

1

PRODUCT OVERVIEW

SAM88RCRI PRODUCT FAMILY

Samsung's SAM88RCRI family of 8-bit single-chip CMOS microcontrollers offers a fast and efficient CPU, a wide range of integrated peripherals, and various mask-programmable ROM sizes.

A dual address/data bus architecture and a large number of bit- or nibble-configurable I/O ports provide a flexible programming environment for applications with varied memory and I/O requirements. Timer/counters with selectable operating modes are included to support real-time operations. Many SAM88RCRI microcontrollers have an external interface that provides access to external memory and other peripheral devices.

KS86C6308/P6308 MICROCONTROLLER

The KS86C6308/P6308 single-chip 8-bit microcontroller is fabricated using an advanced CMOS process. It is built around the powerful SAM88RCRI CPU core.

Stop and Idle power-down modes were implemented to reduce power consumption. To increase on-chip register space, the size of the internal register file was logically expanded. The KS86C6308 has 8 K bytes of program memory on-chip.

Using the SAM88RCRI design approach, the following peripherals were integrated with the SAM88RCRI core:

- Five configurable I/O ports (32 pins)
- 20 bit-programmable pins for external interrupts
- 8-bit timer/counter and 16-bit timer/counter with three operating modes
- Full speed low speed USB function

The KS86C6308/P6308 is a versatile microcontroller that can be used in a wide range of full/low speed USB support general purpose applications. It is especially suitable for use as a keyboard with hub controller and is available in a 64-pin SDIP and a 64-pin QFP package.

OTP

The KS86C6308 microcontroller is also available in OTP (One Time Programmable) version, KS86P6308. KS86P6308 microcontroller has an on-chip 8-Kbyte one-time-programmable EPROM instead of masked ROM. The KS86P6308 is comparable to KS86C6308, both in function and in pin configuration.

FEATURES

CPU

- SAM88RCRI CPU core

Memory

- 8-KB Internal program memory(ROM)
- 256-byte internal register file
(160-byte:General Purpose)

Instruction Set

- 41 instructions
- IDLE and STOP instructions added for power-down modes

Instruction Execution Time

- 332ns at 12 MHz f_{OSC}

Interrupts

- 32 interrupt sources with one vector, each source has its pending bits
- One level, one vector interrupt structure

Oscillation Frequency

- 12 MHz crystal/ceramic oscillator
- External clock source

General I/O

- Bit programmable five I/O ports (30 pins total)

Timer A

- One 8-bit basic timer for watchdog function and programmable oscillation stabilization
programmable 8-bit timer internal generation
function interval, capture, PWM mode
match/capture overflow interrupt

Timer B

- Programmable 16-bit timer interval generation
function interval, capture, PWM mode
match/capture overflow interrupt

Universal Serial Bus with HUB

- 1 upstream port
- 4 downstream port and one embedded function
each port supports separated enable LED built-in 3.3 V voltage regulator

USB/GPIO Function

- Upstream port

Operation Temperature Range

- - 40 °C to + 85 °C

Operation Voltage Range

- 4.0 V to 5.5 V

Package Types

- 64-pin SDIP
- 64-pin QFP

BLOCK DIAGRAM

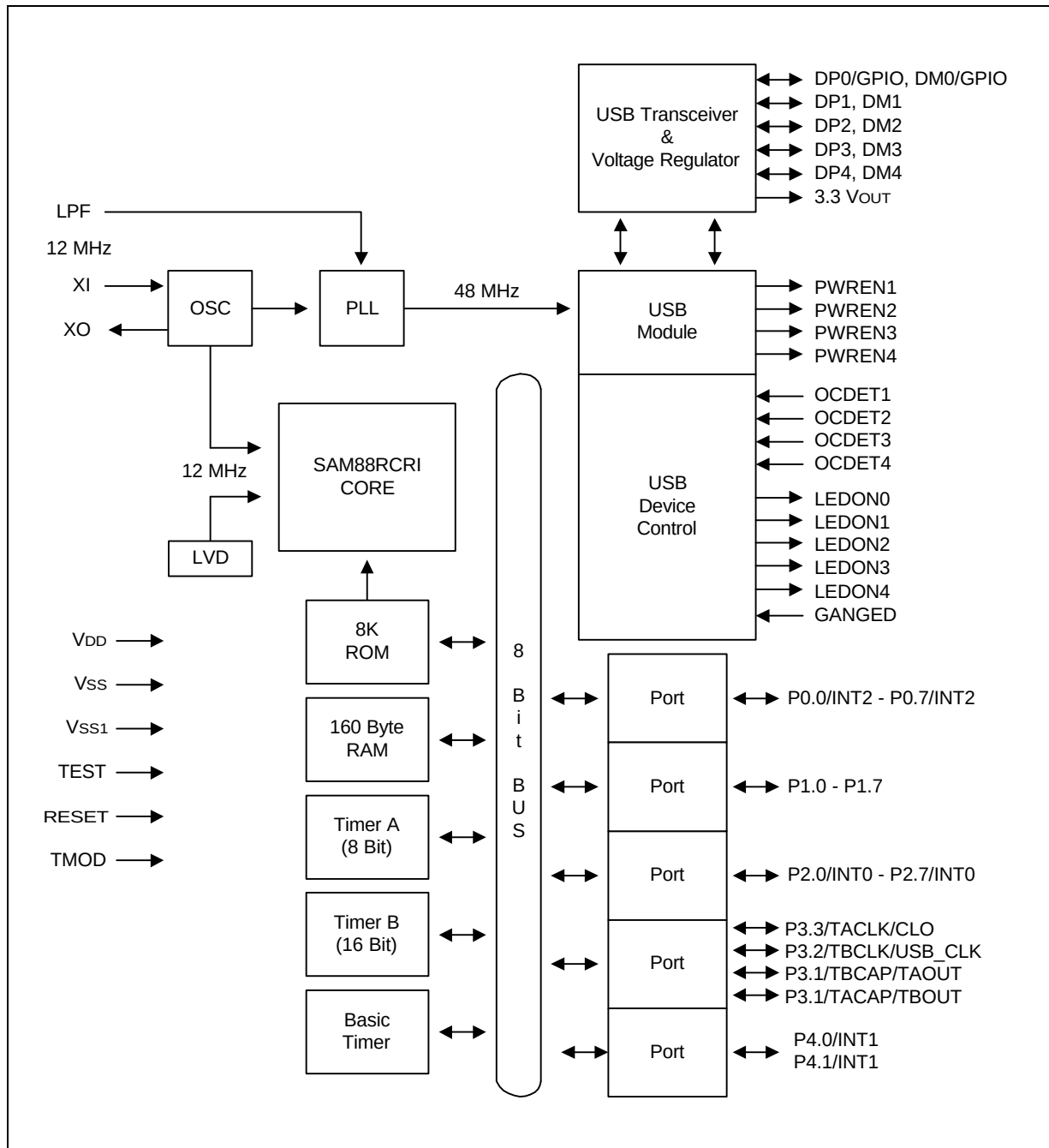


Figure 1-1. Block Diagram

PIN ASSIGNMENTS

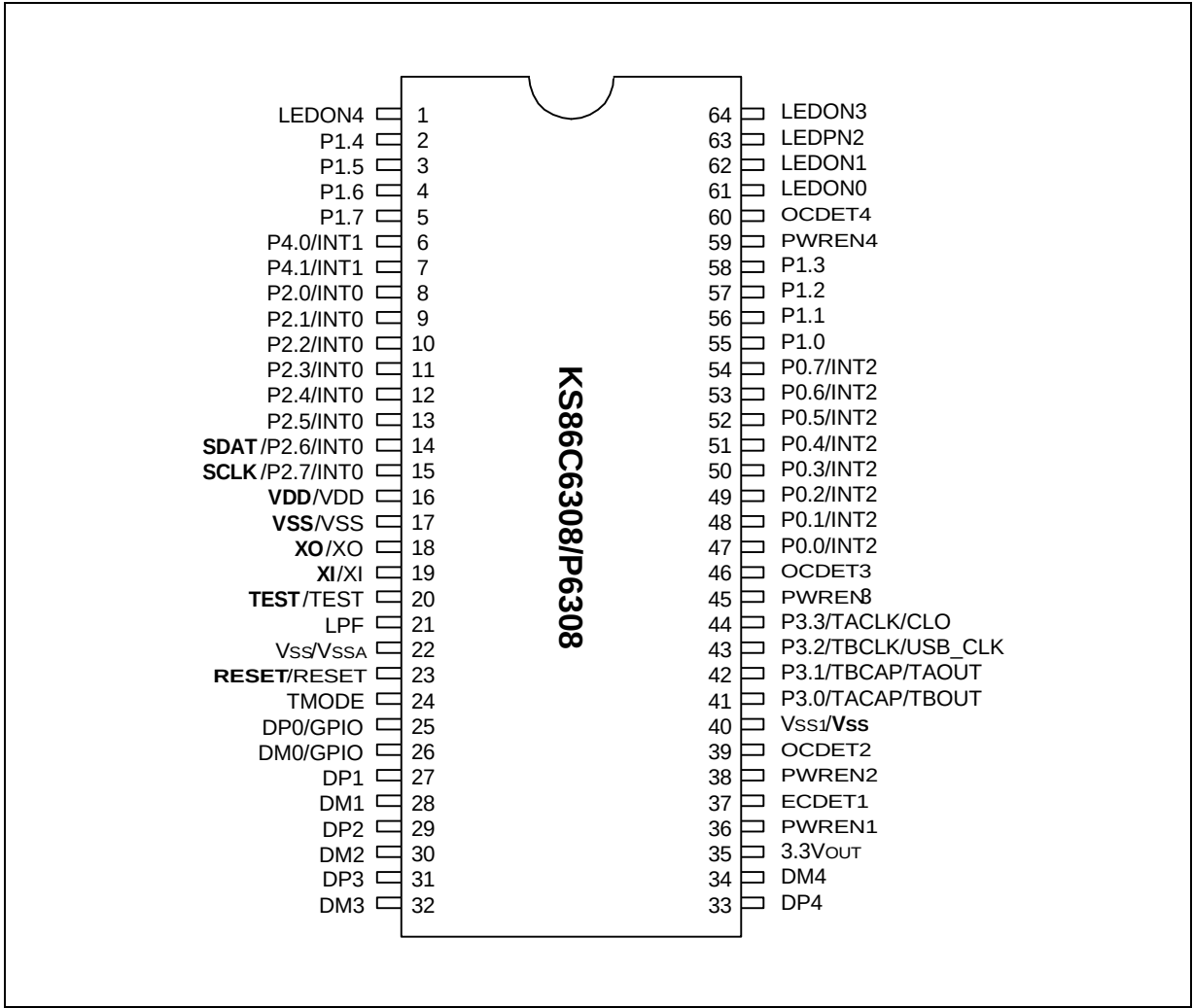


Figure 1-2. Pin Assignment Diagram (64-Pin SDIP Package)

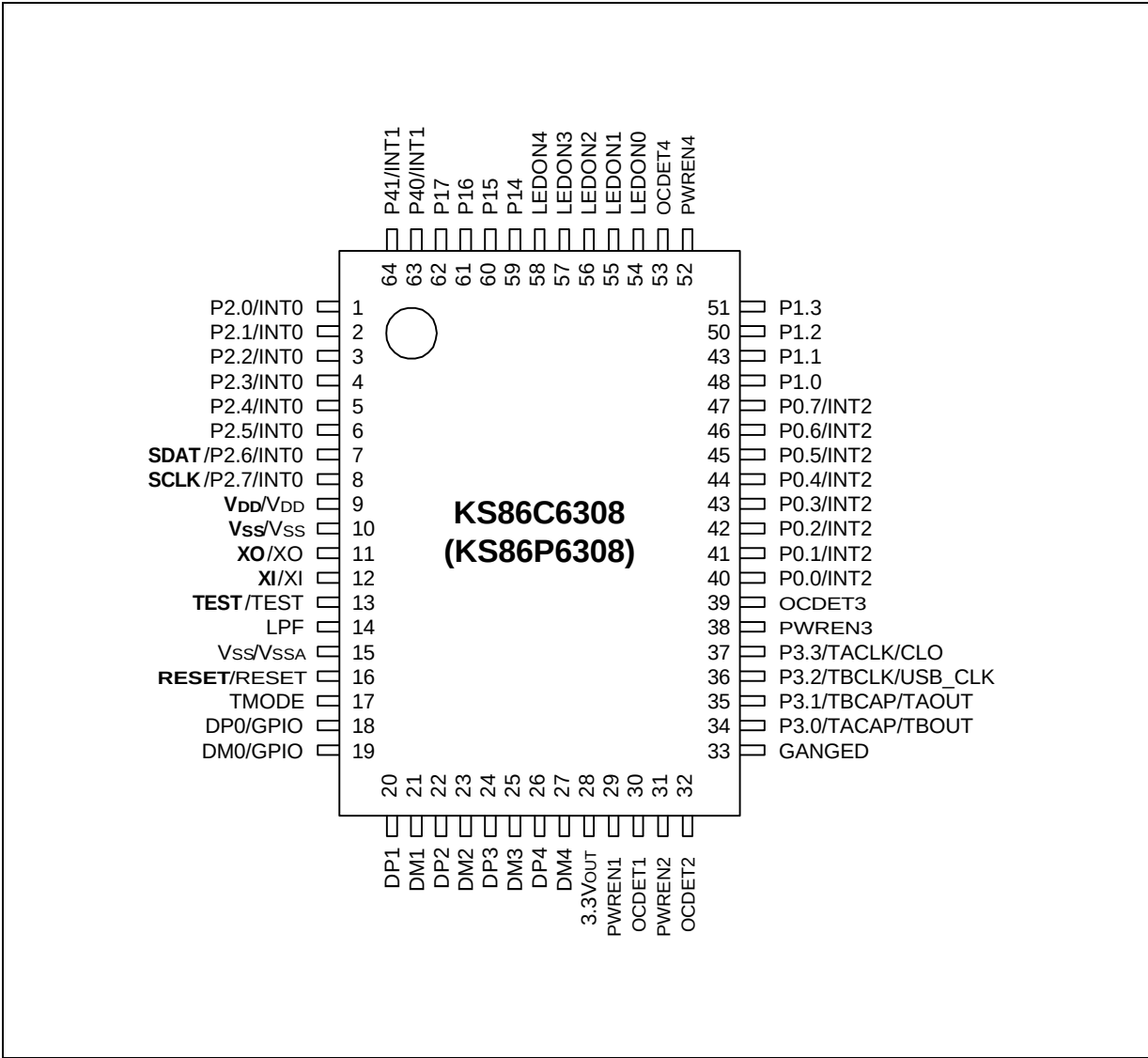


Figure 1-3. Pin Assignment Diagram (64-Pin QFP Package)

PIN DESCRIPTIONS

Table 1-1. KS86C6308/P6308 Pin Descriptions

Pin Names	I/O	Pin Description	Pin Type	Share Pins
P0.0-P0.7	I/O	Bit-programmable I/O port for Schmitt trigger input or open-drain output. Port 0 can be individually configured as external interrupt inputs. Pull-up resistors are assignable by software.	B	INT2
P1.0-P1.7	I/O	Bit-programmable I/O port for Schmitt trigger input or open-drain output. Pull-up resistors are assignable by software.	B	—
P2.0-P2.7	I/O	Bit-programmable I/O port for Schmitt trigger input or open-drain output. Port 2 can also be individually configured as external interrupt inputs. Pull-up resistors are assignable by software.	B	INT0
P3.0-P3.3	I/O	Bit-programmable I/O port for Schmitt trigger input, open-drain output or push-pull output. Port 3 are designed for to drive LED directly. P3.3 can be used to system clock output(CLO) pin. P3.2 PLL clock out for PLL Block.	C	P3.3/TACLK/CLO P3.2/TBCLK/ USB_CLK P3.1/TBCAP/TAOUT P3.0/TACAP/TBOUT
P4.0-P4.1	I/O	Bit-programmable I/O port for Schmitt trigger input or open-drain output or push-pull output. Port4 can also be individually configured as external interrupt inputs. In output mode, pull-up resistors are assignable by software. But in input mode, pull-up resistors are fixed.	D	INT1
3.3 V _{OUT}	—	3.3 V output from internal voltage regulator	—	—
X _{IN} X _{OUT}	—	System clock input and output pin (crystal/ceramic oscillator, or external clock source)	—	—
INT0 INT1 INT2	I	External interrupt for bit-programmable port0, port2 and port4 pins when set to input mode.	—	P2.0-P2.7 P4.0/P4.1 P0.0/P0.7
RESET	I	RESET signal input pin with LVD	A	—
LPF	I	Low Pass Filter Pin for PLL	—	—
TEST	I	Test signal input pin (for factory use only; must be connected to V _{SS})	—	—
TMODE	I	Test signal input pin (for factory use only, must be connected to V _{SS})	—	—
V _{DD}	—	Power input pin	—	—
V _{SS} V _{SS1}	—	VSS1 is a ground power for CPU core. VSS2 is a ground power for I/O and OSC block.	—	—

Table 1-1. KS86C6308/P6308 Pin Descriptions (Continued)

Pin Names	I/O	Pin Description	Pin Type	Share Pins
DP1, DM1 DP2, DM2 DP3, DM3 DP4, DM4	I/O	These pins are an USB Downstream pins.	K	—
DP0/GPIO DM0/GPIO	I/O	These pins are an USB Upstream pin, programmable port for USB interface or General purpose I/O interface.	—	—
LEDON0	O	Root port LED enable. N-channel open-drain output. = 0 Turn LED ON. HUB not Suspend = 1 Turn LED OFF. Reset, Suspend, Transfer in progress	G	—
LEDON1-4	O	Four downstream port LED enable. N-channel open-drain output. = 0 Turn LED ON. Port Enable and HUB not Suspend = 1 Turn LED OFF. Reset, Suspend, Transfer in progress	G	—
OCDET1-4	I	Four downstream power sense = 0 Over Current Detected = 1 Power Okay	F	—
PWREN1-4	O	Power on/off control signals. PWREN1 - PWREN4 are active low, N-CH open-drain outputs. In GANGED mode, all output are swithed together.	G	—
GANGED	I	Gang or Individual Power Control of downstream ports = 0 Individual = 1 Gang	F	—

PIN CIRCUIT DIAGRAMS

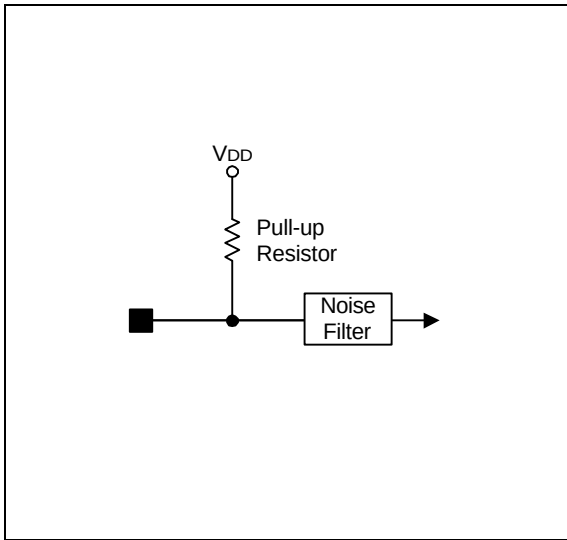


Figure 1-4. Pin Circuit Type A (RESET)

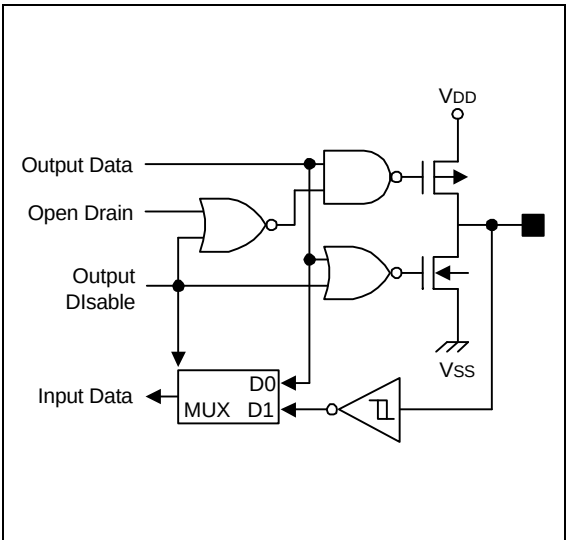


Figure 1-6. Pin Circuit Type C (Port 3)

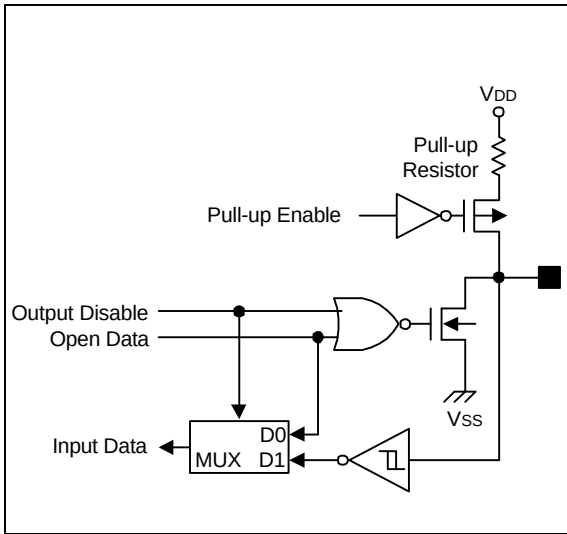


Figure 1-5. Pin Circuit Type B (Port 0, 1, 2)

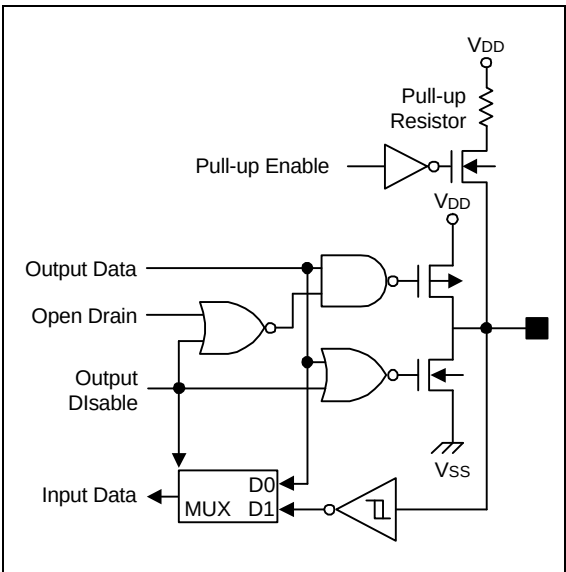


Figure 1-7. Pin Circuit Type D (Port 4)

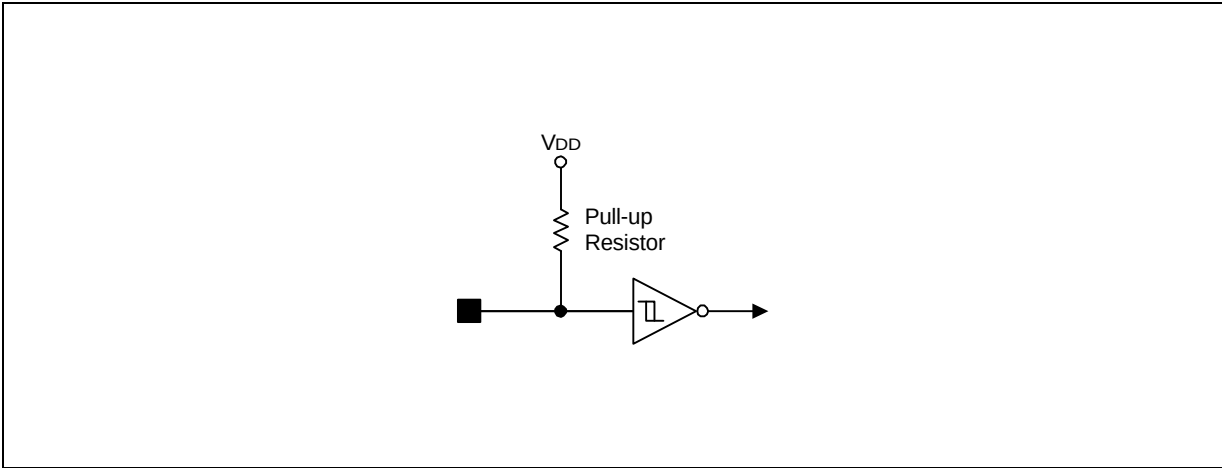


Figure 1-8. Pin Circuit Type F

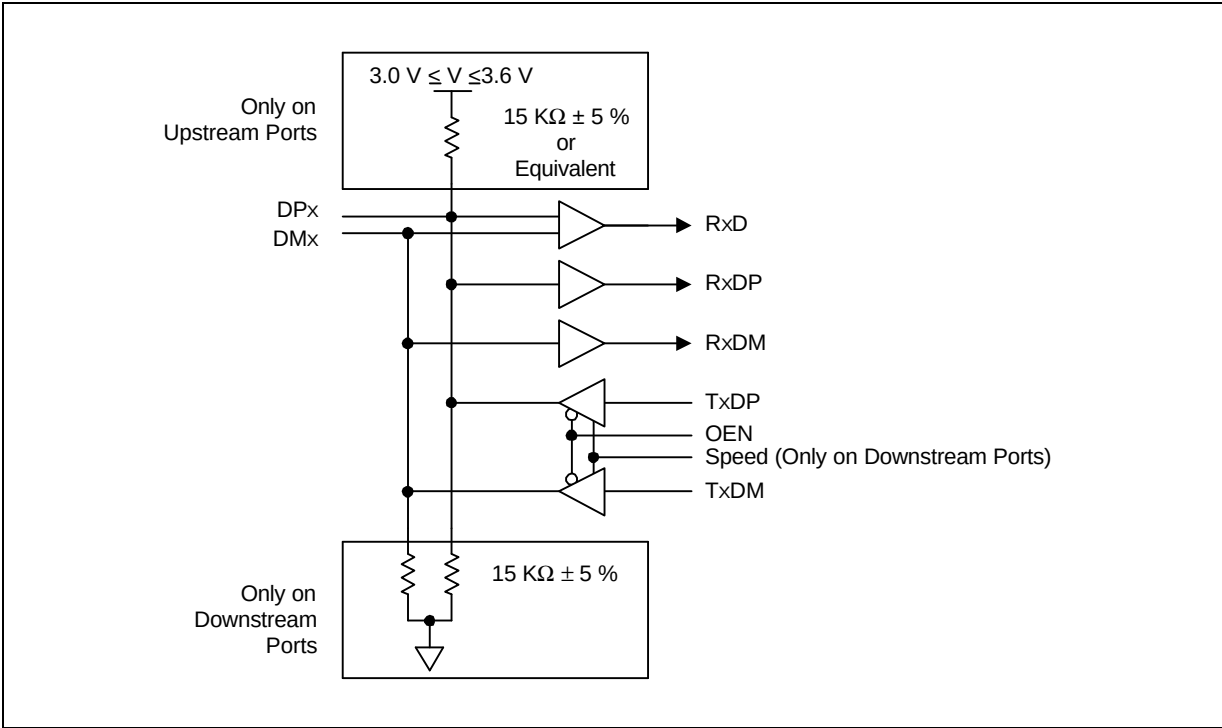


Figure 1-9. Pin Circuit Type K

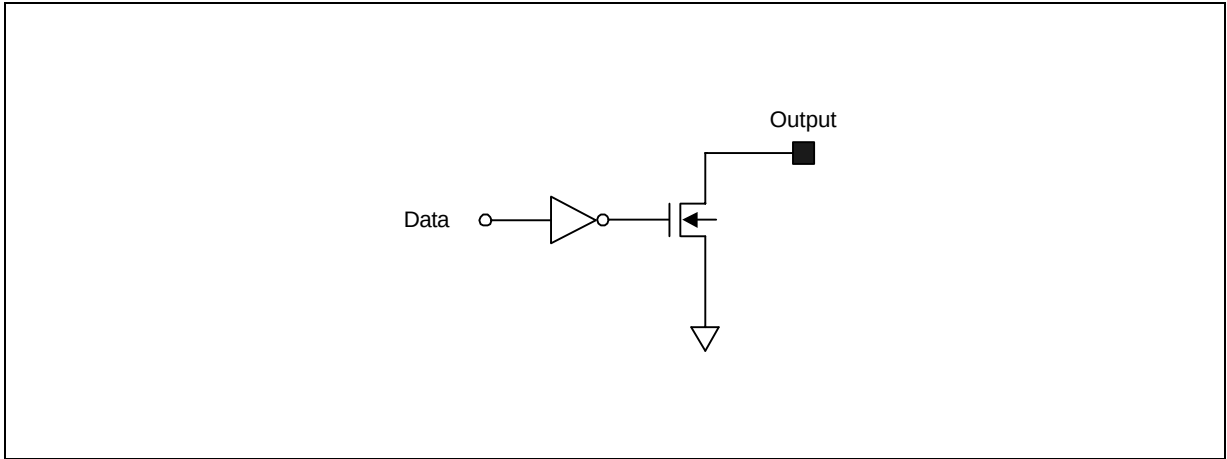


Figure 1-10. Pin Circuit Type G

APPLICATION CITCUIT

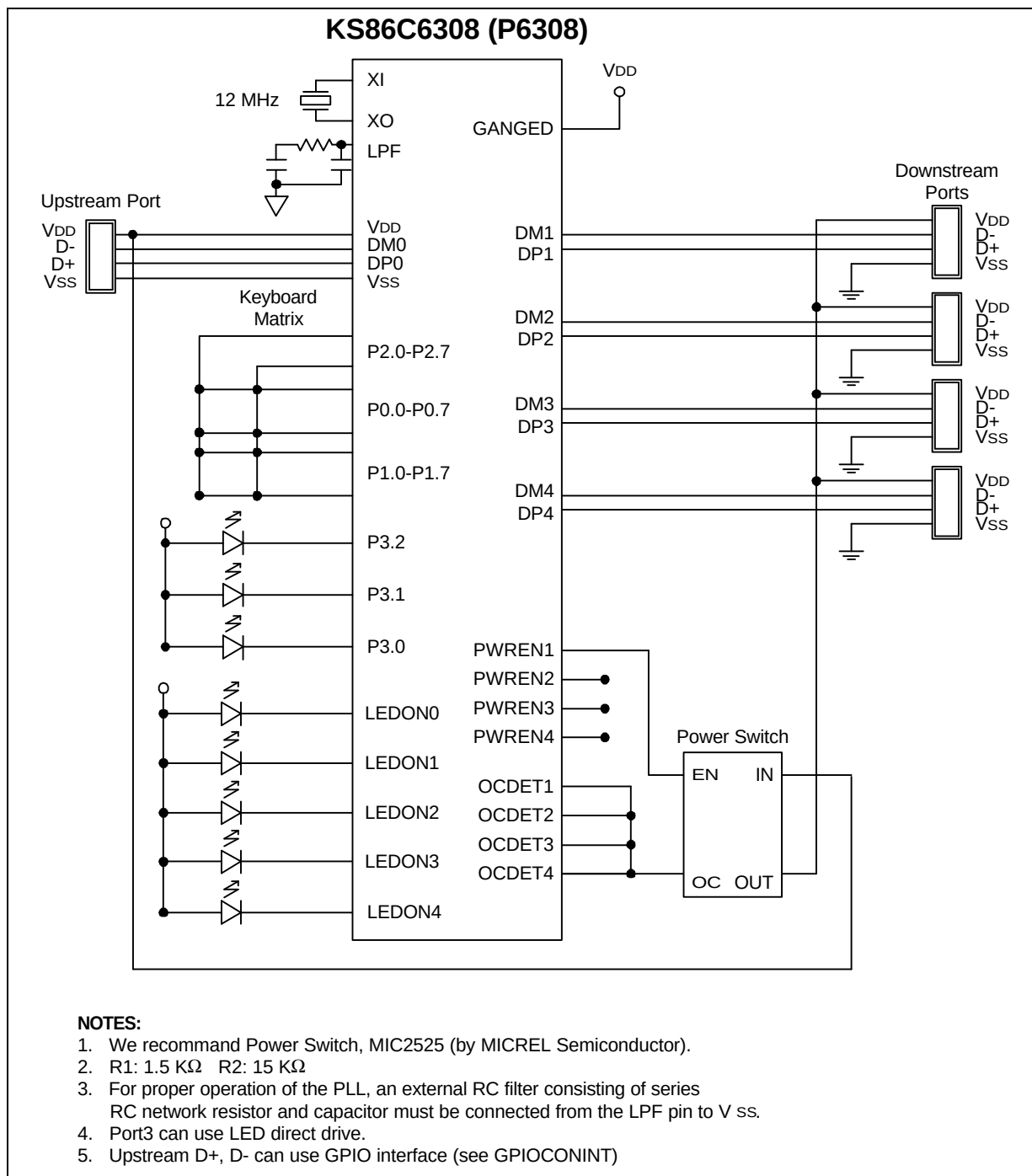


Figure 1-11. Bus-Powered, Gang Port (64-SDIP, 64-QFP)

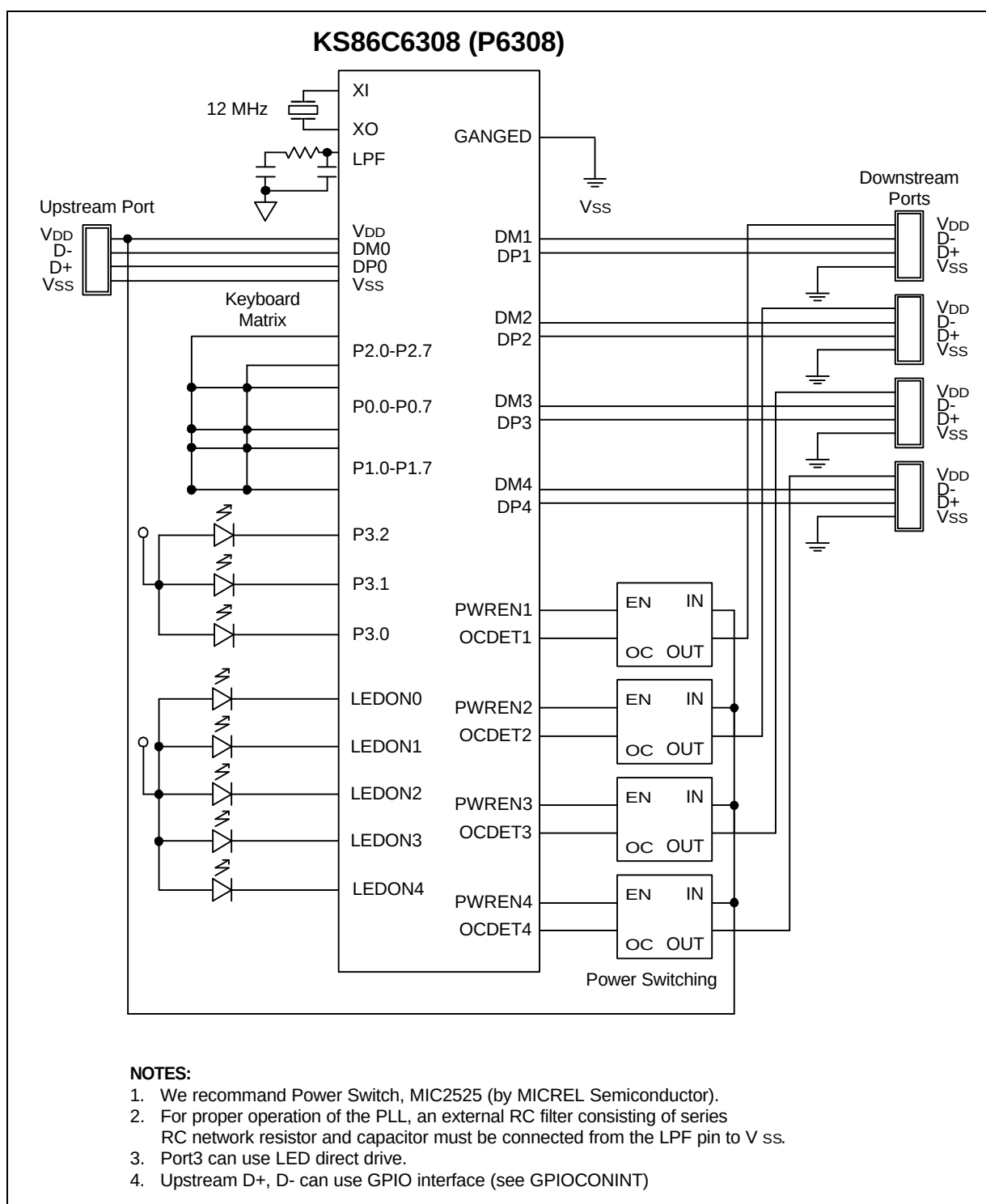


Figure 1-12. Bus-Powered, Individual Port (64-SDIP, 64-QFP)

2

ADDRESS SPACES

OVERVIEW

The KS86C6308/P6308 microcontroller has two kinds of address space:

- Program memory (ROM), internal
- Internal register file

A 13-bit address bus supports both program memory. A separate 8-bit register bus carries addresses and data between the CPU and the internal register file.

The KS86C6308 has 8 K bytes of mask-programmable program memory on-chip. There is one program memory configuration option:

- Internal ROM mode, in which only the 8-Kbyte internal program memory is used.

The KS86C6308/P6308 microcontroller has 160 general-purpose registers in its internal register file. Twenty-seven bytes in the register file are mapped for system and peripheral control functions.

PROGRAM MEMORY (ROM)

Normal Operating Mode (Internal ROM)

The KS86C6308/P6308 has 8 K bytes (locations 0H–1FFFH) of internal mask-programmable program memory.

The first 2 bytes of the ROM (0000H–0001H) are an interrupt vector address.

The program reset address in the ROM is 0100H.

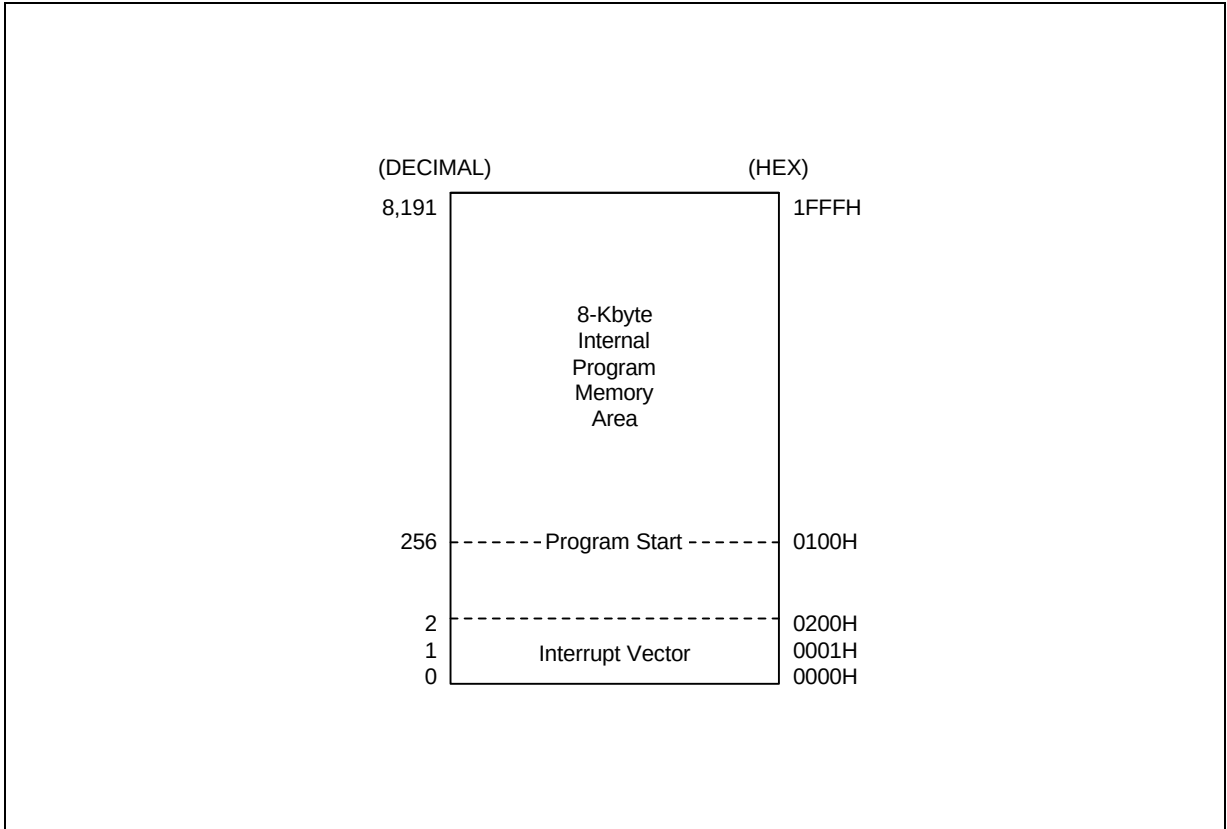


Figure 2-1. Program Memory Address Space

REGISTER ARCHITECTURE

The upper 112 bytes of the KS86C6308/P6308's internal register file are addressed as working registers, system control registers and peripheral control registers. The lower 144 bytes of internal register file (00H–8FH) is called the *general purpose register space*. The total addressable register space is thereby 256 bytes. 160 are available for general-purpose use.

For many SAM88RCRI microcontrollers, the addressable area of the internal register file is further expanded by the additional of one or more register pages at general purpose register space (00H–8FH). This register file expansion is not implemented in the KS86C6308/P6308.

The specific register types and the area (in bytes) that they occupy in the internal register file are summarized in Table 2-1.

Table 2-1. Register Type Summary

Register Type	Number of Bytes
CPU and system control registers	48
Peripheral, I/O, and clock control and data registers	48
General-purpose registers (including the 16-bit common working register area)	160
Total Addressable Bytes	256

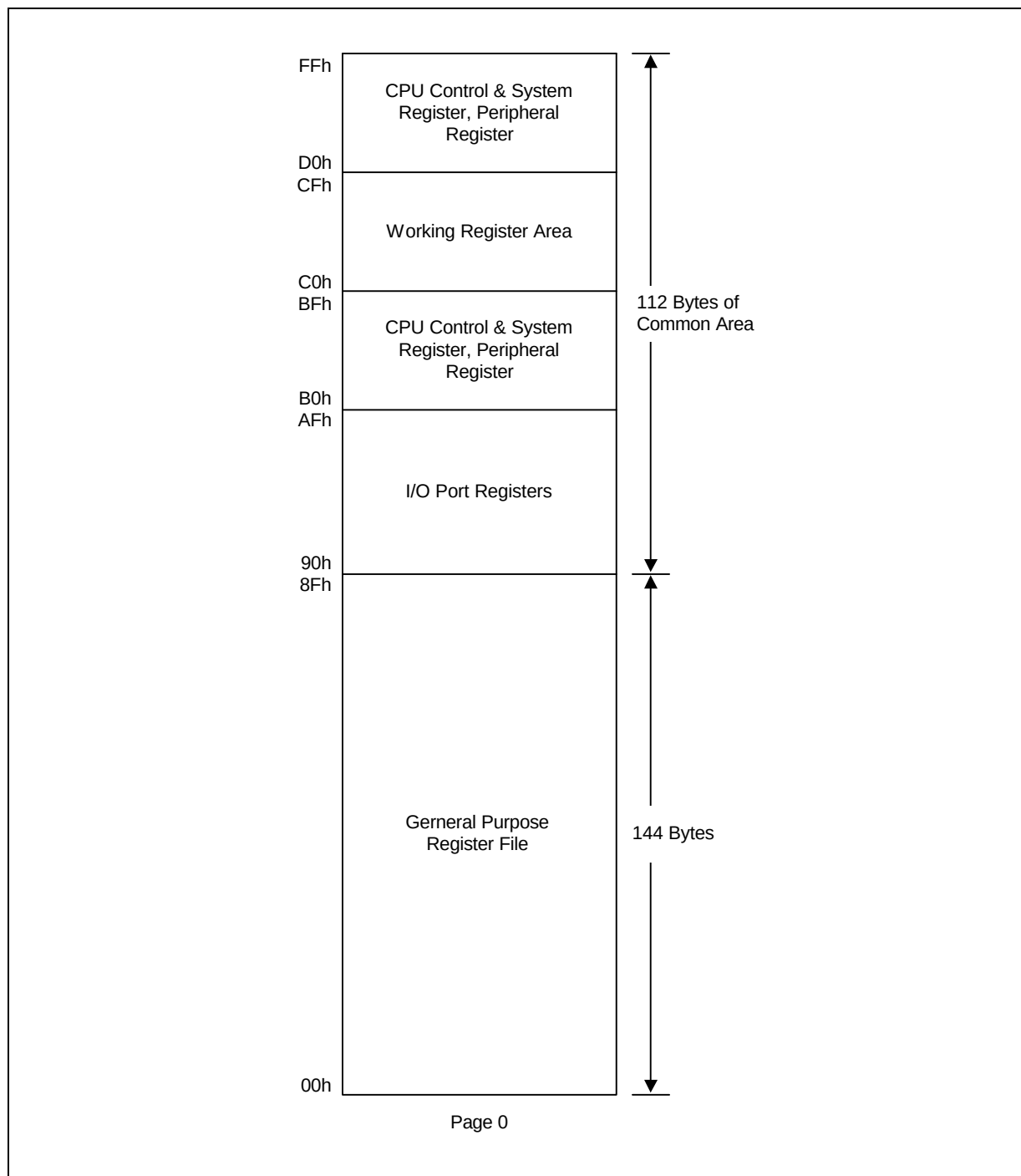


Figure 2-2. Register Architecture

COMMON WORKING REGISTER AREA (C0H–CFH)

The SAM88RCRI register architecture provides an efficient method of working register addressing that takes full advantage of shorter instruction formats to reduce execution time.

This 16-byte address range is called *common area*. That is, locations in this area can be used as working registers by operations that address any location on any page in the register file. Typically, these working registers serve as temporary buffers for data operations between different pages. However, because the KS86C6308/P6308 uses only page 0, you can use the common area for any internal data operation.

The Register (R) addressing mode can be used to access this area

Registers are addressed either as a single 8-bit register or as a paired 16-bit register. In 16-bit register pairs, the address of the first 8-bit register is always an even number and the address of the next register is an odd number. The most significant byte of the 16-bit data is always stored in the even-numbered register; the least significant byte is always stored in the next (+ 1) odd-numbered register.

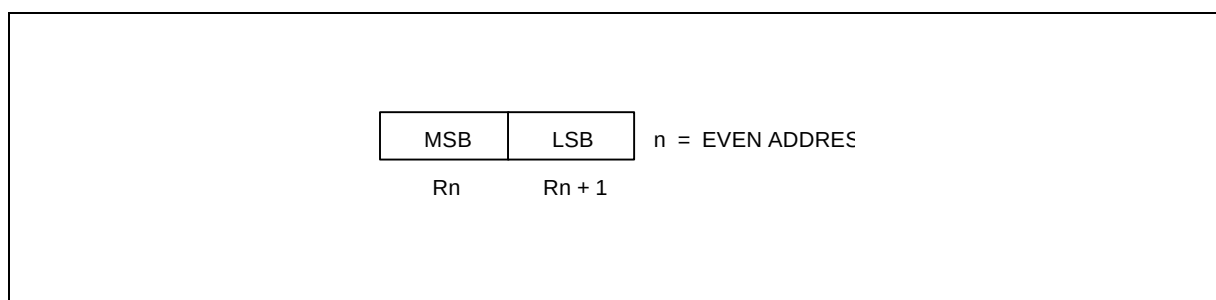


Figure 2-3. 16-Bit Register Pairs

PROGRAMMING TIP — Addressing the Common Working Register Area

As the following examples show, you should access working registers in the common area, locations C0H–CFH, using working register addressing mode only.

- Examples:
1. LD 0C2H,40H ; Invalid addressing mode!
 Use working register addressing instead:
 LD R2,40H ; R2 (C2H) ← the value in location 40H
 2. ADD 0C3H,#45H ; Invalid addressing mode!
 Use working register addressing instead:
 ADD R3,#45H ; R3 (C3H) ← R3 + 45H

SYSTEM STACK

KS86-series microcontrollers use the system stack for subroutine calls and returns and to store data. The PUSH and POP instructions are used to control system stack operations. The KS86C6308/P6308 architecture supports stack operations in the internal register file.

Stack Operations

Return addresses for procedure calls and interrupts and data are stored on the stack. The contents of the PC are saved to stack by a CALL instruction and restored by the RET instruction. When an interrupt occurs, the contents of the PC and the FLAGS register are pushed to the stack. The IRET instruction then pops these values back to their original locations. The stack address is always decremented *before* a push operation and incremented *after* a pop operation. The stack pointer (SP) always points to the stack frame stored on the top of the stack, as shown in Figure 2-4.

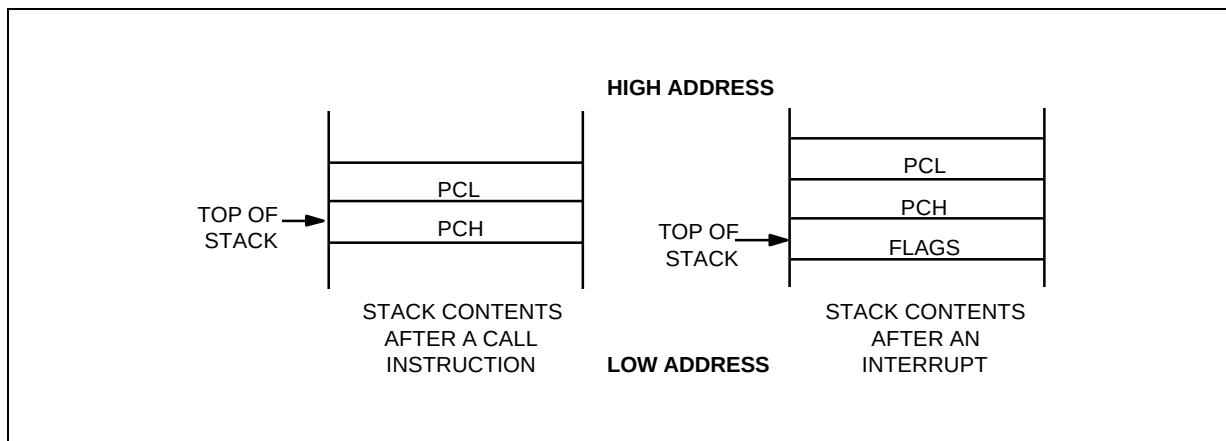


Figure 2-4. Stack Operations

Stack Pointer (SP)

Register location D9H contains the 8-bit stack pointer (SP) that is used for system stack operations. After a reset, the SP value is undetermined.

Because only internal memory space is implemented in the KS86C6308/P6308, the SP must be initialized to an 8-bit value in the range 00H–8FH.

NOTE: In case a Stack Pointer is initialized to 00H, it is decreased to FFH when stack operation starts. This means that a Stack Pointer access invalid stack area.

PROGRAMMING TIP — Standard Stack Operations Using PUSH and POP

The following example shows you how to perform stack operations in the internal register file using PUSH and POP instructions:

```

LD      SP,#0C0H      ; SP = C0H (Normally, the SP is set to 0C0H by the
                      ; initialization routine)
.
.
.
PUSH    SYM            ; Stack address 0BFH = SYM
PUSH    CLKCON         ; Stack address 0BEH = CLKCON
PUSH    20H            ; Stack address 0BDH = 20H
PUSH    R3             ; Stack address 0BCH = R3
.
.
.
POP     R3             ; R3 = Stack address 0BCH
POP     20H            ; 20H = Stack address 0BDH
POP     CLKCON         ; CLKCON = Stack address 0BEH
POP     SYM            ; SYM = Stack address 0BFH

```

NOTES

3

ADDRESSING MODES

OVERVIEW

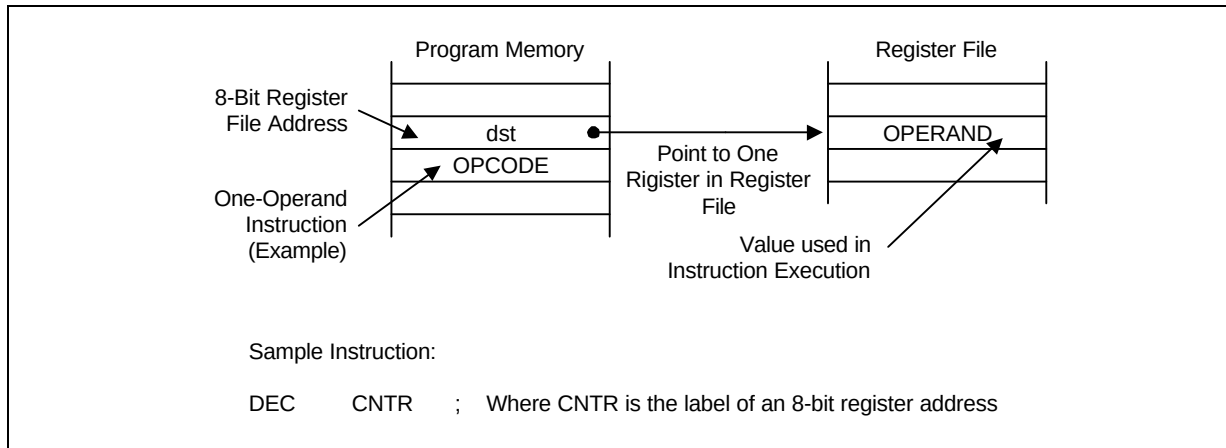
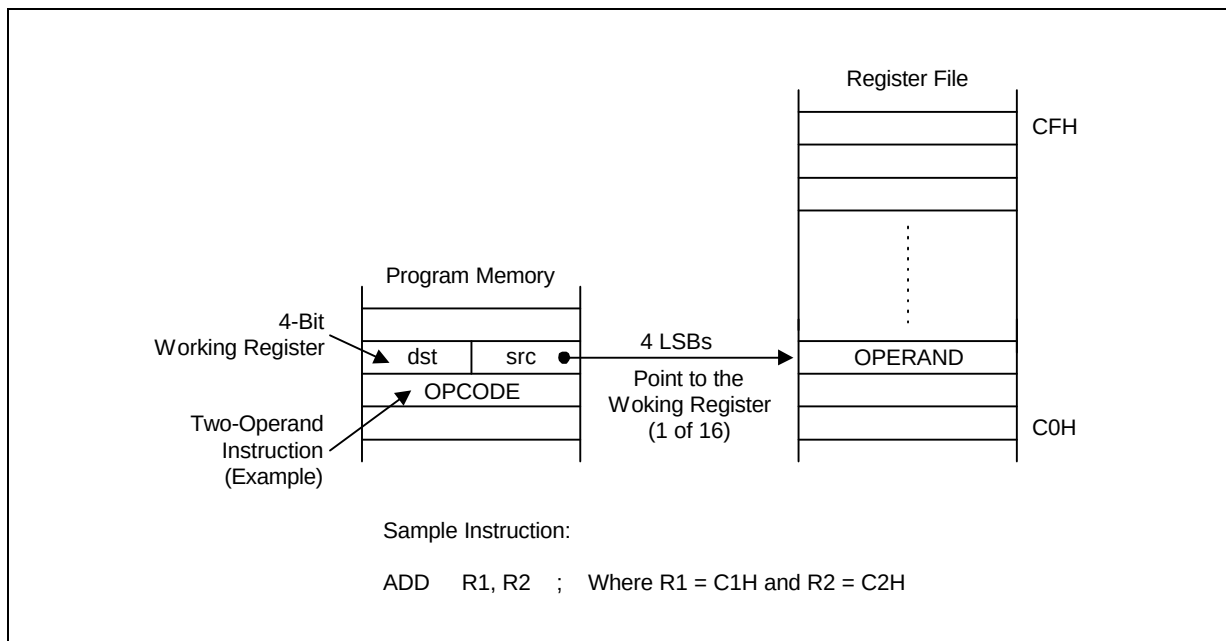
Instructions that are stored in program memory are fetched for execution using the program counter. Instructions indicate the operation to be performed and the data to be operated on. Addressing mode is the method used to determine the location of the data operand. The operands specified in SAM88RCRI instructions may be condition codes, immediate data, or a location in the register file, program memory, or data memory.

The SAM88RCRI instruction set supports six explicit addressing modes. Not all of these addressing modes are available for each instruction. The addressing modes and their symbols are as follows:

- Register (R)
- Indirect Register (IR)
- Indexed (X)
- Direct Address (DA)
- Relative Address (RA)
- Immediate (IM)

REGISTER ADDRESSING MODE (R)

In Register addressing mode, the operand is the content of a specified register (see Figure 3-1). Working register addressing differs from Register addressing because it uses an 16-byte working register space in the register file and an 4-bit register within that space (see Figure 3-2).

**Figure 3-1. Register Addressing****Figure 3-2. Working Register Addressing**

INDIRECT REGISTER ADDRESSING MODE (IR)

In Indirect Register (IR) addressing mode, the content of the specified register or register pair is the address of the operand. Depending on the instruction used, the actual address may point to a register in the register file, to program memory (ROM), or to an external memory space (see Figures 3-3 through 3-6).

You can use any 8-bit register to indirectly address another register. Any 16-bit register pair can be used to indirectly address another memory location.

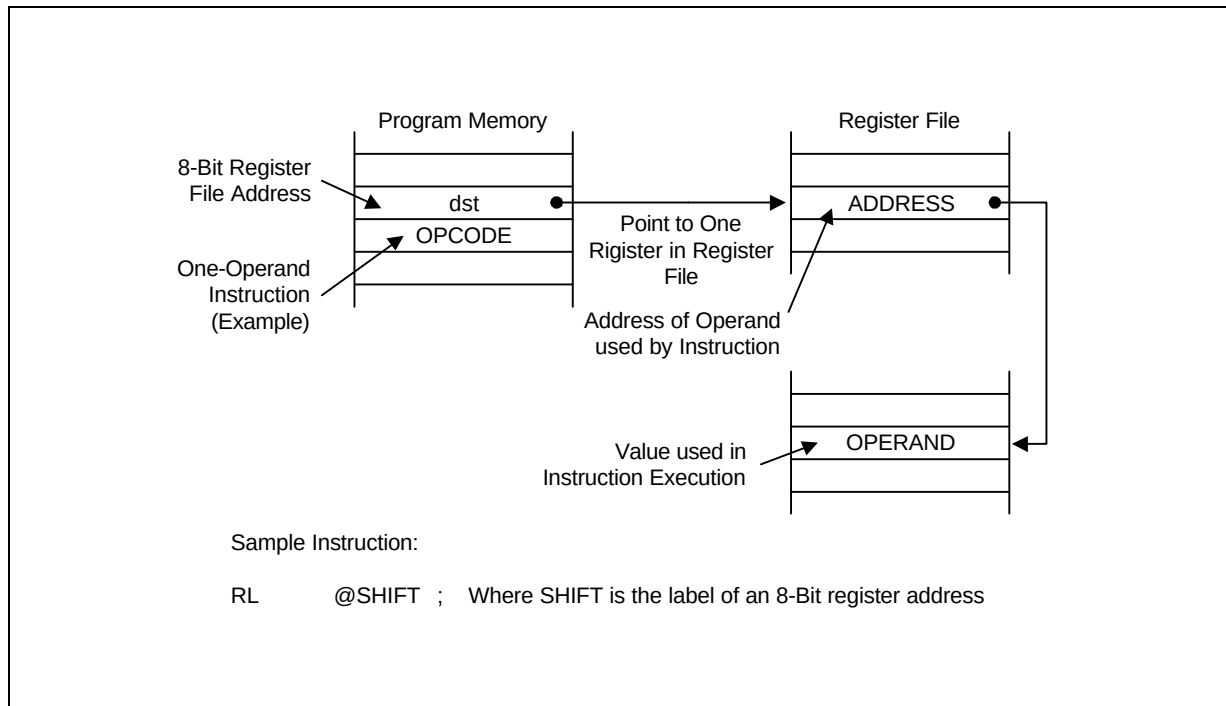


Figure 3-3. Indirect Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Continued)

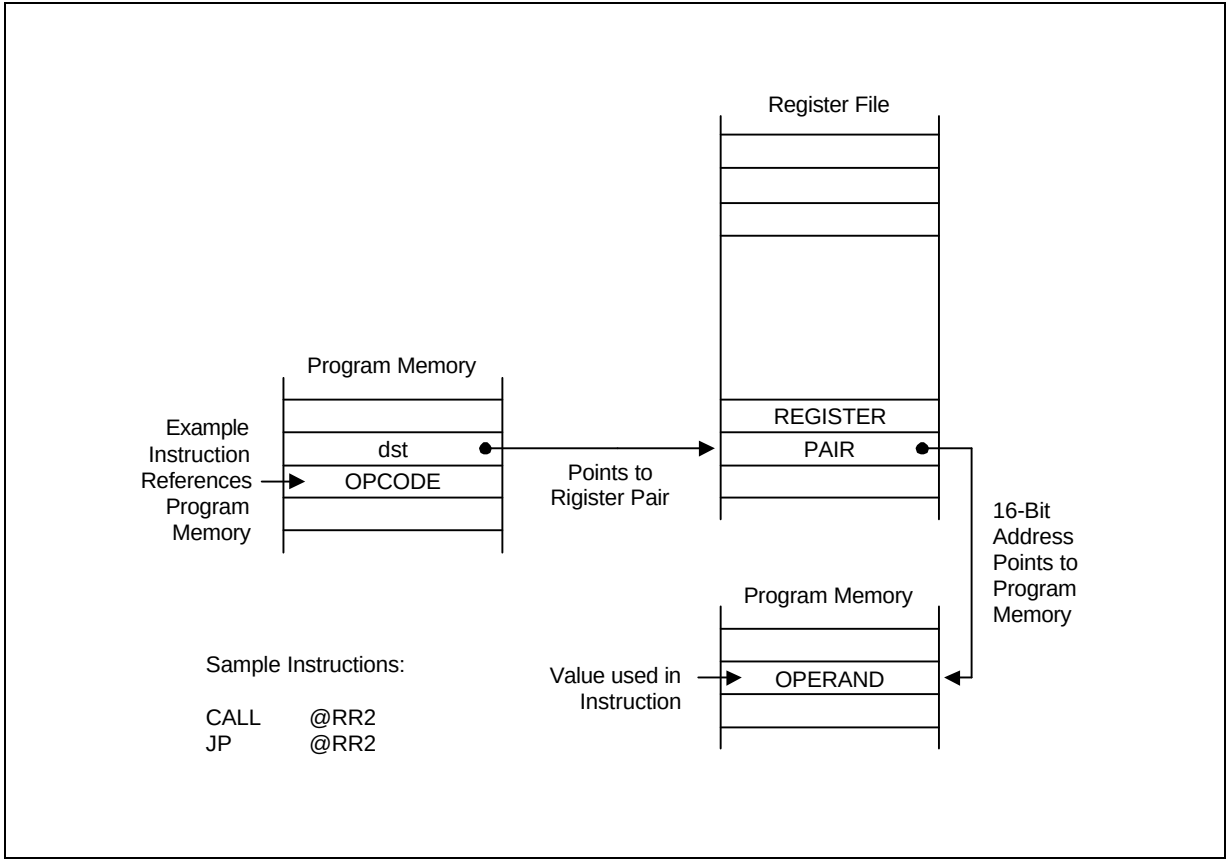


Figure 3-4. Indirect Register Addressing to Program Memory

INDIRECT REGISTER ADDRESSING MODE (Continued)

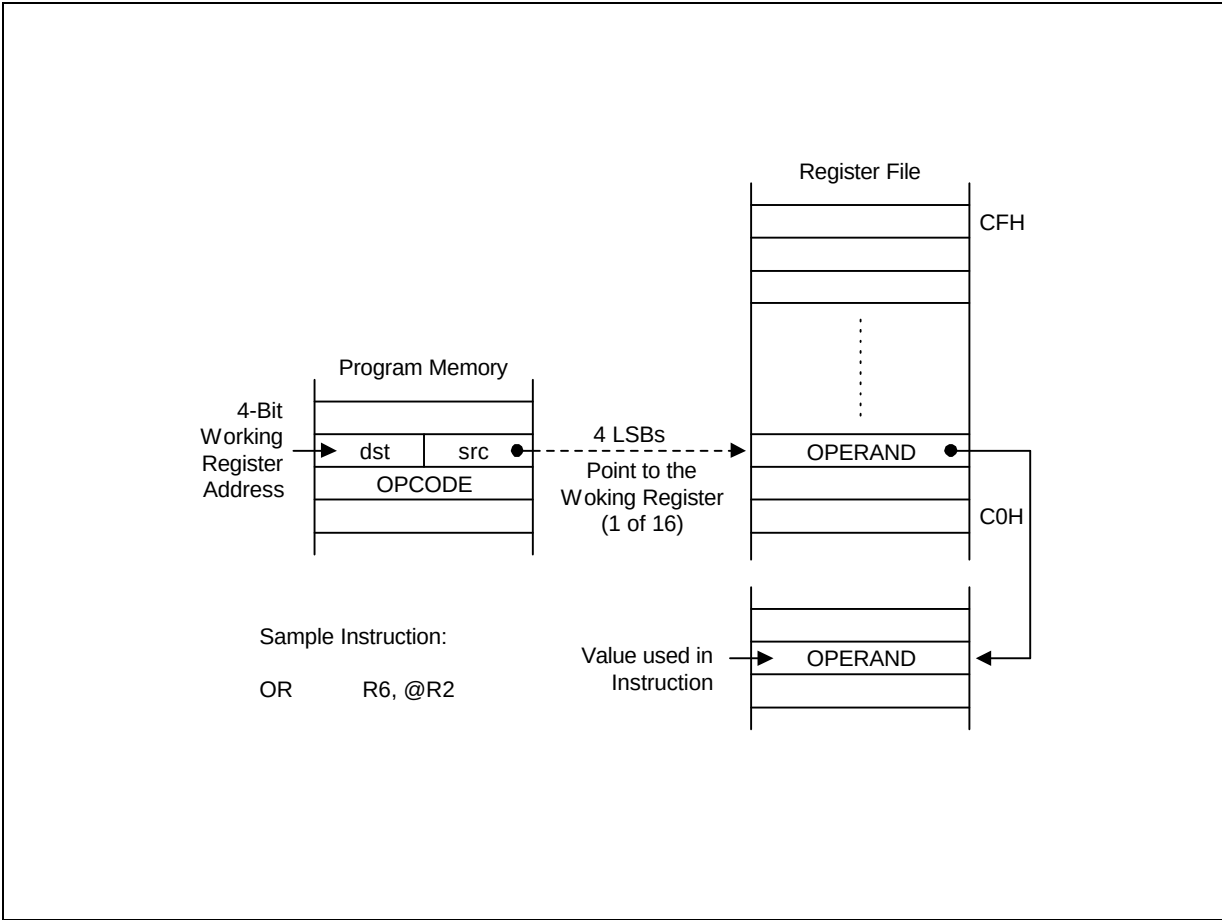


Figure 3-5. Indirect Working Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Concluded)

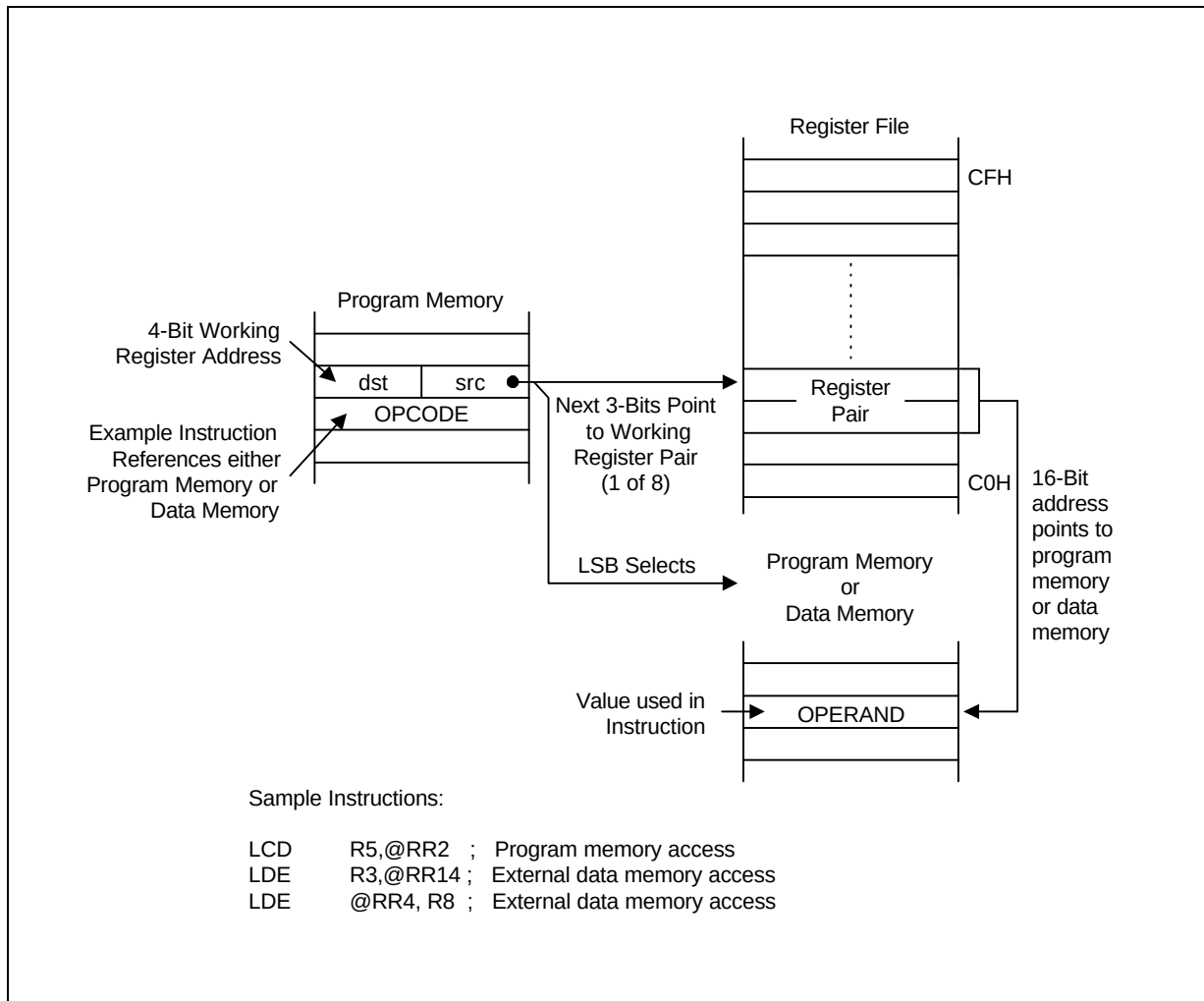


Figure 3-6. Indirect Working Register Addressing to Program or Data Memory

INDEXED ADDRESSING MODE (X)

Indexed (X) addressing mode adds an offset value to a base address during instruction execution in order to calculate the effective operand address (see Figure 3-7). You can use Indexed addressing mode to access locations in the internal register file or in external memory.

In short offset Indexed addressing mode, the 8-bit displacement is treated as a signed integer in the range -128 to +127. This applies to external memory accesses only (see Figure 3-8).

For register file addressing, an 8-bit base address provided by the instruction is added to an 8-bit offset contained in a working register. For external memory accesses, the base address is stored in the working register pair designated in the instruction. The 8-bit or 16-bit offset given in the instruction is then added to the base address (see Figure 3-9).

The only instruction that supports Indexed addressing mode for the internal register file is the Load instruction (LD). The LDC and LDE instructions support Indexed addressing mode for internal program memory, external program memory, and for external data memory, when implemented.

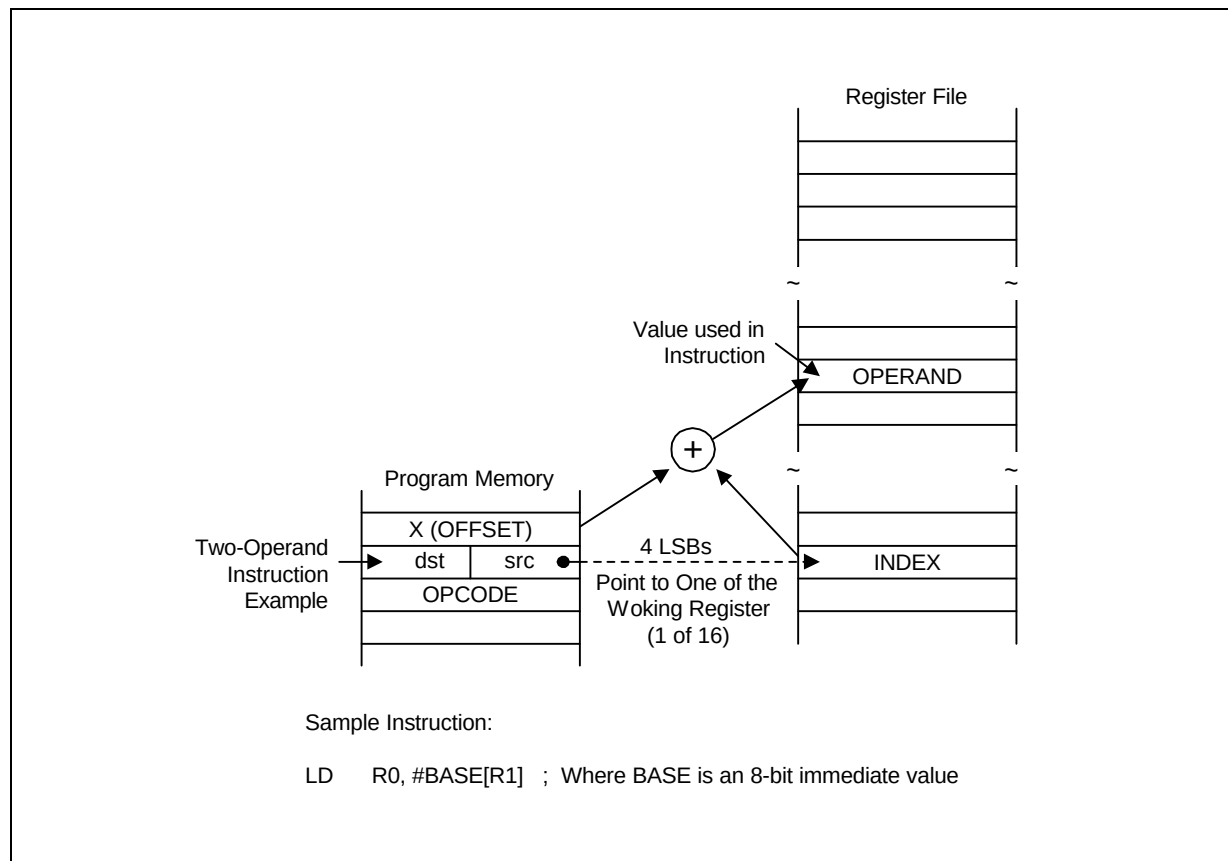


Figure 3-7. Indexed Addressing to Register File

INDEXED ADDRESSING MODE (Continued)

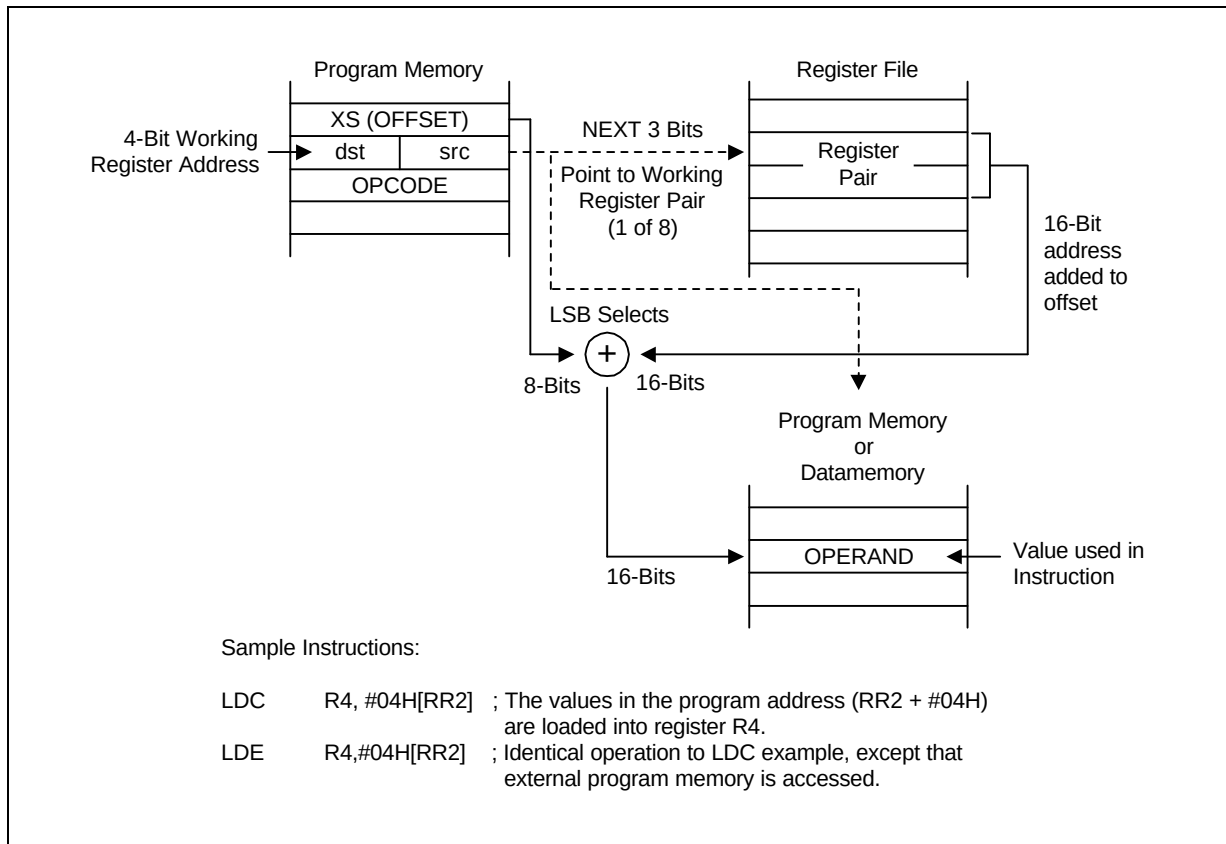


Figure 3-8. Indexed Addressing to Program or Data Memory with Short Offset

INDEXED ADDRESSING MODE (Concluded)

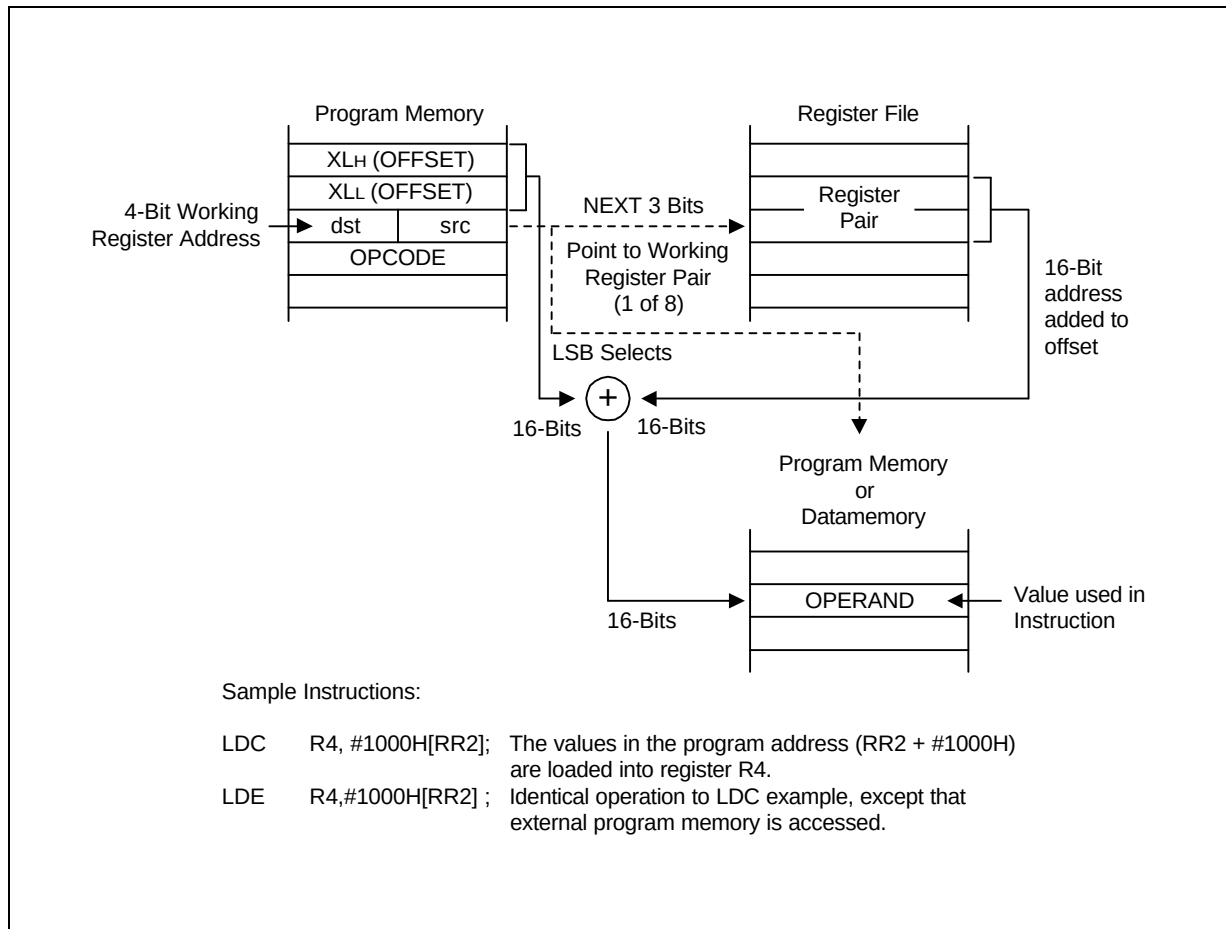


Figure 3-9. Indexed Addressing to Program or Data Memory with Long Offset

DIRECT ADDRESS MODE (DA)

In Direct Address (DA) mode, the instruction provides the operand's 16-bit memory address. Jump (JP) and Call (CALL) instructions use this addressing mode to specify the 16-bit destination address that is loaded into the PC whenever a JP or CALL instruction is executed.

The LDC and LDE instructions can use Direct Address mode to specify the source or destination address for Load operations to program memory (LDC) or to external data memory (LDE), if implemented.

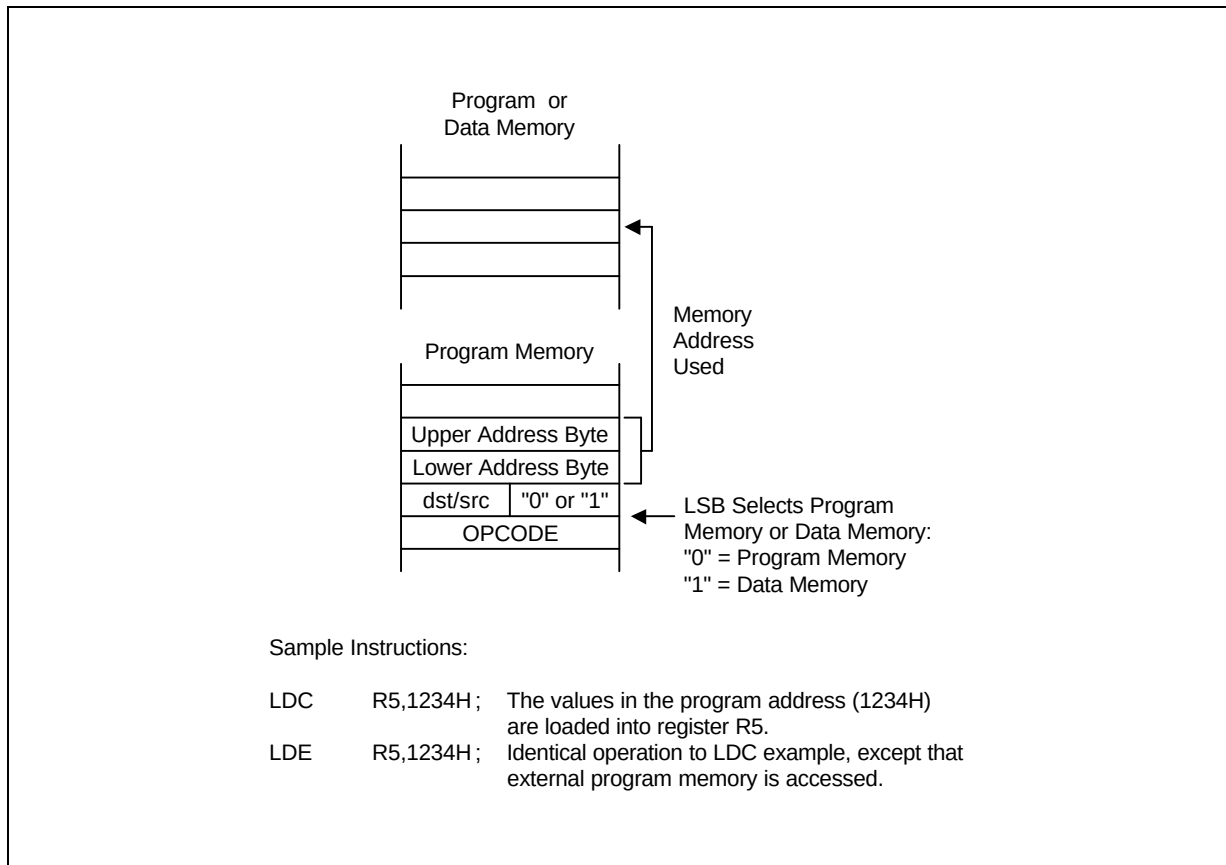


Figure 3-10. Direct Addressing for Load Instructions

DIRECT ADDRESS MODE (Continued)

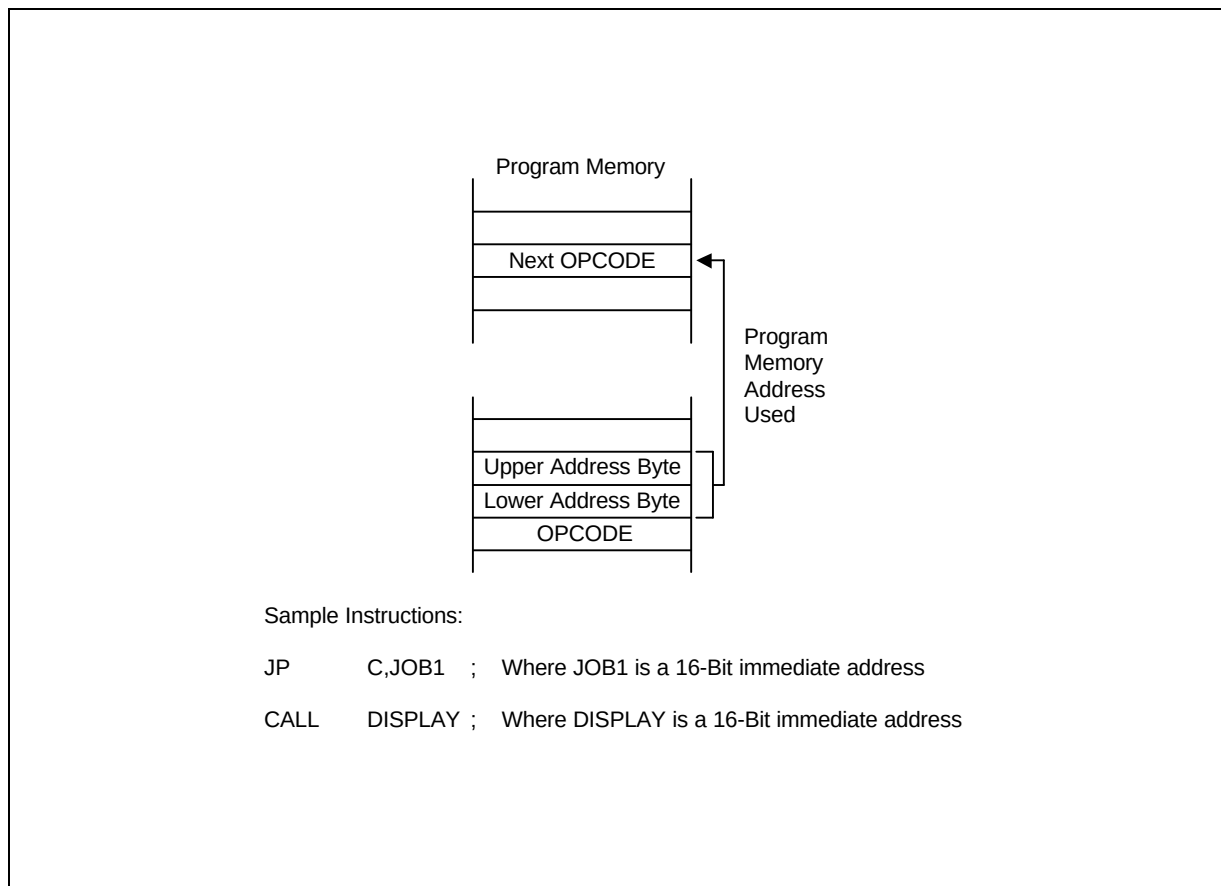


Figure 3-11. Direct Addressing for Call and Jump Instructions

RELATIVE ADDRESS MODE (RA)

In Relative Address (RA) mode, a two's-complement signed displacement between -128 and $+127$ is specified in the instruction. The displacement value is then added to the current PC value. The result is the address of the next instruction to be executed. Before this addition occurs, the PC contains the address of the instruction immediately following the current instruction.

The instructions that support RA addressing is JR.

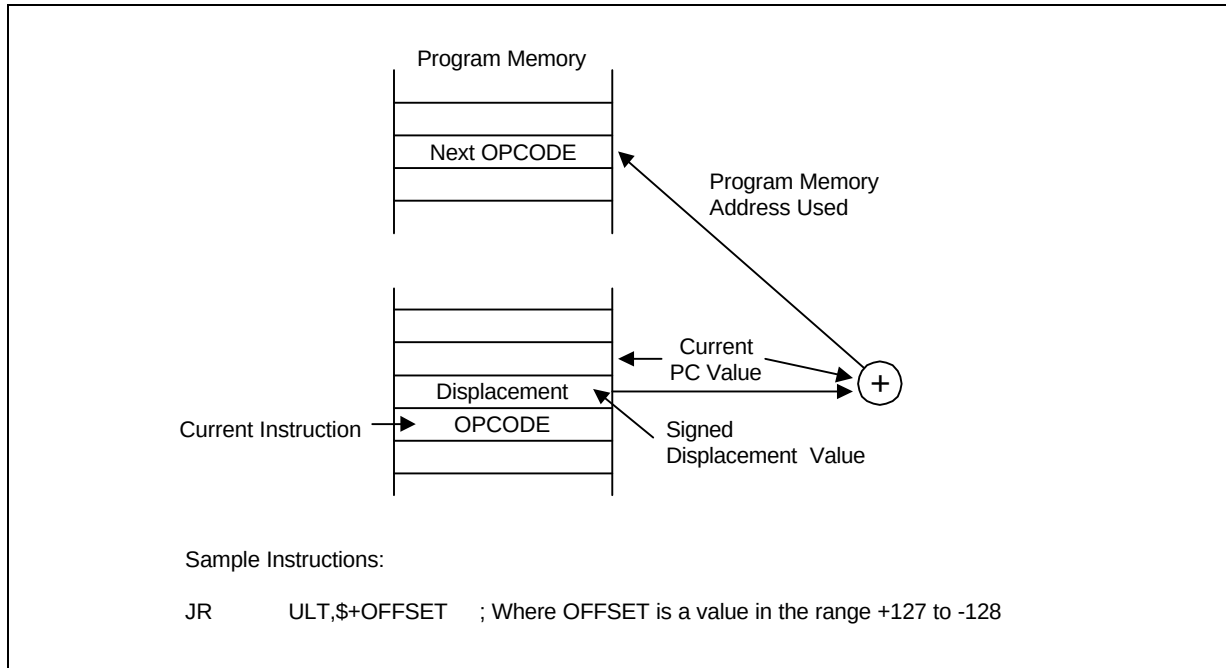


Figure 3-12. Relative Addressing

IMMEDIATE MODE (IM)

In Immediate (IM) addressing mode, the operand value used in the instruction is the value supplied in the operand field itself. Immediate addressing mode is useful for loading constant values into registers.

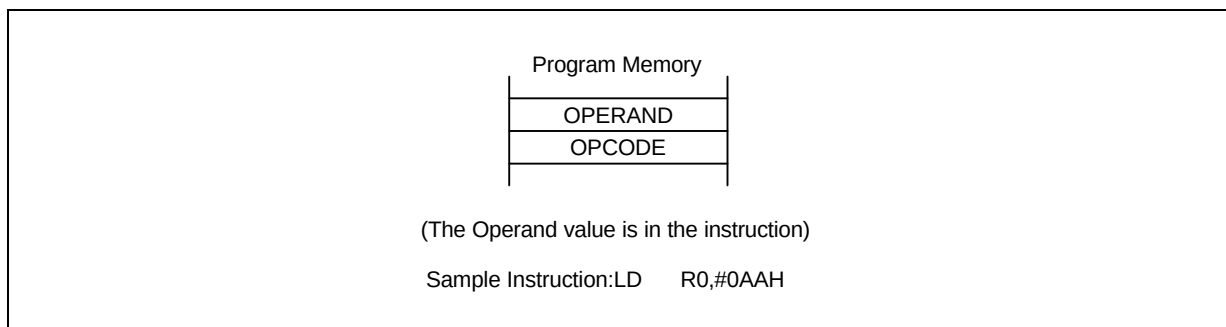


Figure 3-13. Immediate Addressing

4

CONTROL REGISTERS

OVERVIEW

In this section, detailed descriptions of the KS86C6308/P6308 control registers are presented in an easy-to-read format. These descriptions will help familiarize you with the mapped locations in the register file. You can also use them as a quick-reference source when writing application programs.

System and peripheral registers are summarized in Table 4-1. Figure 4-1 illustrates the important features of the standard register description format.

Control register descriptions are arranged in alphabetical order according to register mnemonic. More information about control registers is presented in the context of the various peripheral hardware descriptions in Part II of this manual.

Table 4-1. System and Peripheral control Registers

Register Name	Mnemonic	Decimal	Hex	R/W
Timer A counter register	TACNT	208	D0H	R
Timer A data register	TADATA	209	D1H	R/W
Timer A control register	TACON	210	D2H	R/W
Location D3H is not mapped.				
Clock control register	CLKCON	212	D4H	R/W
System flags register	FLAGS	213	D5H	R/W
Locations D6H-D8H are not mapped.				
Stack pointer	SP	217	D9H	R/W
Port 0 interrupt pending register	P0PND	218	DAH	R/W
Location DBH is not mapped.				
Basic timer control register	BTCON	220	DCH	R/W
Basic timer counter register	BTCNT	221	DDH	R
Location DEH is not mapped.				
System mode register	SYM	223	DFH	R/W
Port 0 data register	P0	224	E0H	R/W
Port 1 data register	P1	225	E1H	R/W
Status Change Endpoint	STSCHGEP	226	E2H	R/W
Miscellaneous Control Register	MISCCON	227	E3H	R/W
Suspend/Resume Control Register	SUSRECON	228	E4H	R/W
Port Bus State Port selected by PORTSEL register Bit0=VMIN Bit1=VPIN Bit7-Bit2=0	PBUSSTE	229	E5H	R/W
HUB Interface Status LSB	HINTFSTS0	230	E6H	R/W
HUB Interface Status MSB	HINTFSTS1	231	E7H	R/W
Full Speed Tranceiver crossover point control	FSXCON	232	E8H	R/W
Low Speed Tranceiver crossover point control	LSXCON	233	E9H	R/W
HUB Interrupt Status register	HINTSTS	234	EAH	R/W
HUB Interrupt Enable register	HINTEN	235	EBH	R/W
Embebedded Function Change Status register. Similar to embedded port register. Refer to explanation ablvec	EMBFCSTS	236	ECH	R/W
Embebedded Function Status register, it is similar to embedded port register but this is for MCU, while embedded port register is for the HOST	EMBFSTS	237	EDH	R/W

Table 4-1. System and Peripheral control Registers (Continued)

Register Name	Mnemonic	Decimal	Hex	R/W
Device Status LSB	DSTS0	238	EEH	R/W
Device Status MSB	DSTS1	239	EFH	R/W
HUB Endpoint1 Status LSB	HEP1STS0	240	F0H	R/W
HUB Endpoint1 Status MSB	HEP1STS1	241	F1H	R
HUB Endpoint0 Status LSB	HEP0STS0	242	F2H	R/W
HUB Endpoint0 Status MSB	HEP0STS0	243	F3H	W
HUB Status register LSB	HSTS0	244	F4H	R/W
HUB Status register MSB	HSTS1	245	F5H	R
HUB Change status register LSB	HSTS2	246	F6H	R/W
HUB Change status register MSB	HSTS3	247	F7H	R/W
HUB Endpoint Byte count register indicates bytes present in the OUT FIFO	HEP0BCNT	248	F8H	W
HUB Endpoint0 data buffer	HEP0BUF	249	F9H	R/W
HUB Endpoint0 control/status register	HEP0CSR	250	FAH	R/W
Port select, selects port for status transfer =1 downstream port1 =2 downstream port2 =3 downstream port3 =4 downstream port4	PORTSEL	251	FBH	R/W
Port Status LSB	PXSTS0	252	FCH	R
Port Status MSB	PXSTS1	253	FDH	R
Port Status Change LSB	PXSTS2	254	FEH	R
Port Status Change MSB	PXSTS3	255	FFH	R
Lower 8 bits of the Frame Num. The frame number register contains the frame number received with the last SOF	FRAMEL	176	B0H	R
Upper bits of the Frame Num.	FRAMEH	177	B1H	R
Interrupt Pending register	INTPEND	178	B2H	R/W
Interrupt Mask Register	INTMASK	179	B3H	R/W
Function endpoint0 Data buffer	FEP0BUF	180	B4H	R/W
Function endpoint1 Data buffer	FEP1BUF	181	B5H	R/W
Function endpoint2 Data buffer	FEP2BUF	182	B6H	R/W
Function Interrupt Status register	FINTSTS	183	B7H	R/W
Function Interrupt Enable register	FINTEN	184	B8H	R/W

Table 4-1. System and Peripheral control Registers (Concluded)

Register Name	Mnemonic	Decimal	Hex	R/W
Function endpoint byte count register, valid for transfer	FEPBCNT	185	B9H	R
Endpoint In Control/Status register	FEPINCSR	186	BAH	R/W
Endpoint Out Control/Status register	FEPOUTCSR	187	BBH	R/W
Endpoint Configuration register	FEPCFG	188	BCH	R/W
Endpoint Max Packet size register	FEPMAXP	189	BDH	R/W
Endpoint Select register	FEPSEL	190	BEH	R/W
Function USB Address register	FADDR	191	BFH	R/W
Port0 control register(LOW byte)	P0CONL	160	A0H	R/W
Port0 control register(HIGH byte)	P0CONH	161	A1H	R/W
Port1 control register(LOW byte)	P1CONL	162	A2H	R/W
Port1 control register(HIGH byte)	P1OCNH	163	A3H	R/W
Port2 control register(LOW byte)	P2CONL	164	A4H	R/W
Port2 control register(HIGH byte)	P2CONH	165	A5H	R/W
Port0 interrupt control register	P0INT	166	A6H	R/W
Port0 interrupt pending register	P0PND	167	A7H	R/W
Port2 interrupt control register	P2INT	168	A8H	R/W
Port2 interrupt pending register	P2PND	169	A9H	R/W
Port3 Control register	P3CON	170	AAH	R/W
Port4 Control register	P4CON	171	ABH	R/W
Port4 interrupt enable & pending register	P4INTn P4PNDn	172	ACH	R/W
Port2 Data register	P2	173	ADH	R/W
Port3 Data register	P3	174	AEH	R/W
Port4 Data register	P4	175	AFH	R/W
Not mapped 90H-95H				
USB mode select register	USBMODE	150	96H	R/W
USB/GPIO mode select register	USBSEL	151	97H	R/W
DM0/GPIO,DP0/GPIO Data register (Only GPIO mode)	GPIODATA	152	98H	R/W
GPIO control and interrupt pending register	GPIOCONINT	153	99H	R/W
Timer A counter register High	TACNTH	154	9AH	R
Timer A counter register Low	TACNTL	155	9BH	R
Timer A Data register High	TADATAH	156	9CH	R/W
Timer A Data register Low	TADATAL	157	9DH	R/W
Timer A Control register	TACON	158	9EH	R/W

Timer interrupt priority pending register	INTPND	159	9FH	R/W
---	--------	-----	-----	-----

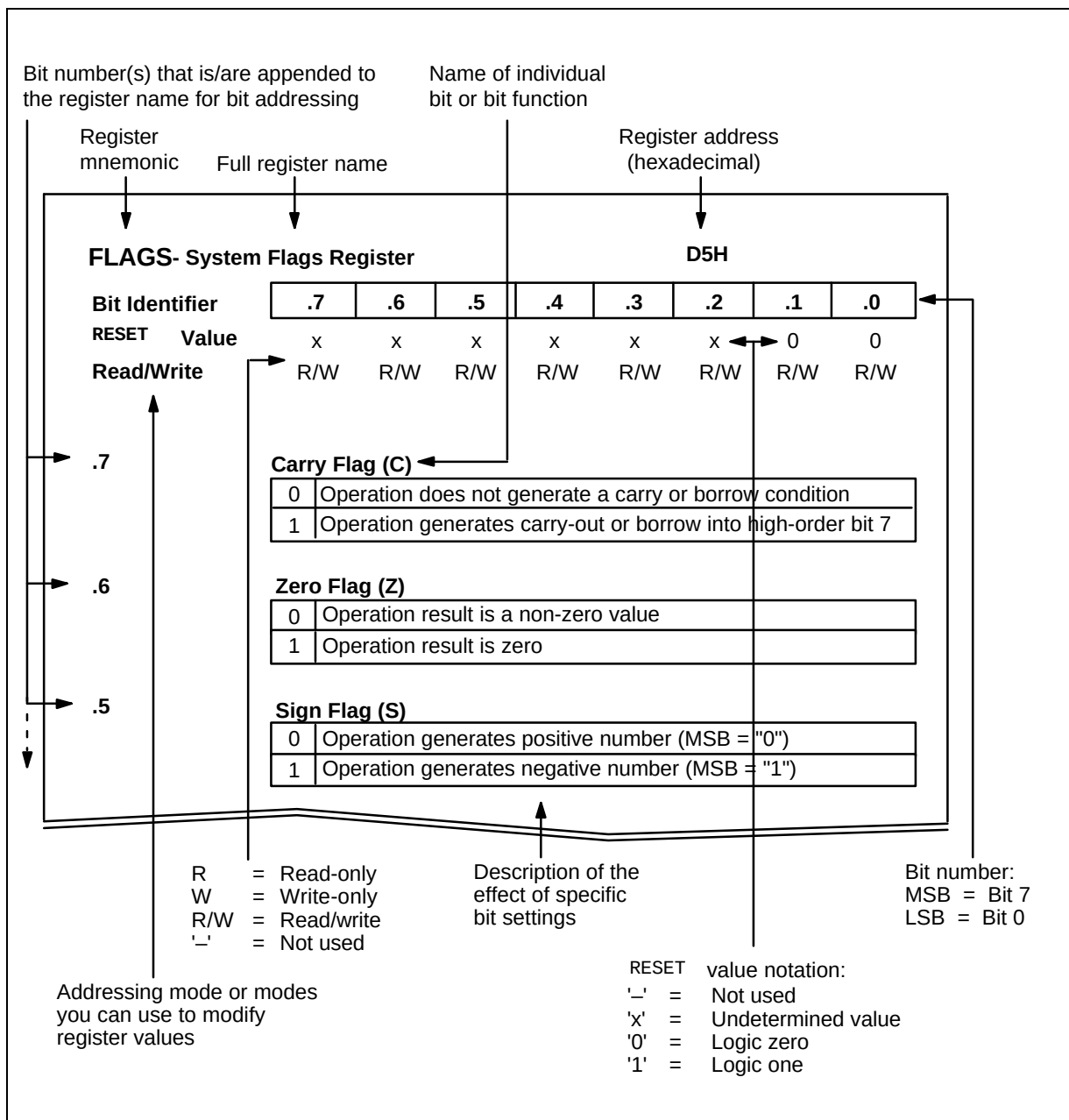


Figure 4-1. Register Description Format

BTCON — Basic Timer Control Register**DCH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7-.4**Watchdog Timer Enable Bits**

1	0	1	0	Disable watchdog function
Any other value				Enable watchdog function

.3 and .2**Basic Timer Input Clock Selection Bits**

0	0	$f_{OSC}/4096$
0	1	$f_{OSC}/1024$
1	0	$f_{OSC}/128$
1	1	Invalid setting

.1**Basic Timer Counter Clear Bit ^(note)**

0	No effect
1	Clear BTCNT

.0**Basic Timer Divider Clear Bit ^(note)**

0	No effect
1	Clear both dividers

NOTE: When you write a "1" to BTCON.0 (or BTCON.1), the basic timer counter (or basic timer divider) is cleared. The bit is then cleared automatically to "0".

CLKCON — System Clock Control Register**D4H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7**Oscillator IRQ Wake-up Function Bit**

0	Enable IRQ for main system oscillator wake-up in power down mode
1	Disable IRQ for main system oscillator wake-up in power down mode

.6 and .5

Not used for KS86C6308/P6308

.4 and .3**CPU Clock (System Clock) Selection Bits ⁽¹⁾**

0	0	Divide by 16 ($f_{OSC}/16$)
0	1	Divide by 8 ($f_{OSC}/8$)
1	0	Divide by 2 ($f_{OSC}/2$)
1	1	Non-divided clock (f_{OSC}) ⁽²⁾

.2-.0

Not used for KS86C6308/P6308

NOTES:

- After a reset, the slowest clock (divided by 16) is selected as the system clock. To select faster clock speeds, load the appropriate values to CLKCON.3 and CLKCON.4.
- f_{OSC} means oscillator frequency.

EMBFCSTS— Embedded Function Status Change Register**ECH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	0	–	0	–	–
Read/Write	–	–	–	R/W	–	R/W	–	–

.7 - .5	Not used for KS86C6308/P6308
---------	------------------------------

.4	C_RST Bit	
	0	No effect
	1	Reset change interrupt pending (when this bit is read)
	1	Set to clear pending (when this bit is written)

.3	Not used for KS86C6308/P6308
----	------------------------------

.2	C_SUSP Bit	
	0	No effect
	1	Reset change interrupt pending (when this bit is read)
	1	Set to clear pending (when this bit is written)

.1 - .0	Not used for KS86C6308/P6308
---------	------------------------------

EMBFSTS — Embedded Function Status Register**EDH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	0	–	0	0	0
Read/Write	–	–	–	R/W	–	R	R	R/W

.7 - .5	Not used for KS86C6308/P6308
---------	------------------------------

.4	Function RESET Bit
0	Cleared by the HUB when RESET bit in the embedded port register has been cleared.
1	Set by the host to selectively reset the function

.3	Not used for KS86C6308/P6308
----	------------------------------

.2	Function SUSPEND Bit
0	Cleared by the host or when MCU sends remote resume.
1	Set by the host to selectively suspend embedded function.

.1	Function Enable Bit
0	Function is disabled
1	Function is enabled

.0	Function Connection Bit
0	Cleared by the hardware when detecting USB RESET from the host or POWER ON RESET .
1	Set by the MCU to indicate to the HUB controller that embedded function is present and ready

FADDR — Function Address Register**BFH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	0	0	0	0	0	0	0
Read/Write	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7	Not used for KS86C6308/P6308
----	------------------------------

.6 - .0	Function Address Bits (Loads the address received)
---------	---

Note: This register should be loaded before setting DATAEND and clearing OUTPKTRDY in the EP0CSR.

FEPBCNT — Function Endpoint Byte Count Register**B9H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	0	0	0	0	0
Read/Write	–	–	–	R/W	R/W	R/W	R/W	R/W

.7 - .5	Not used for KS86C6308/P6308
---------	------------------------------

.4 - .0	Byte Count Bits
	Contains byte count of FIFO

Byte Count: Once the MCU detects OUT_PKT_RDY set in CSR, then it can read this register to find out the number of bytes to be from Endpoint FIFO.

FEPCFG — Function Endpoint Configuration Register**BCH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	—	—	—	—
Read/Write	R/W	R/W	R/W	R/W	—	—	—	—

.7	Transfer Direction Bit	
0	Out, host to endpoint	
1	In, endpoint to host	

.6	OUT Endpoint Mode Selection Bit	
0	Interrupt or Bulk	
1	ISO	

.5	IN Endpoint Mode Selection Bit	
0	Interrupt or Bulk	
1	ISO	

.4	IN Endpoint ISO Update Mode Status Bit	
0	No effect	
1	When this bit is set, a packet written to IN endpoint is sent to the host only after SOF is received.	

.3 - .0	Not used for KS86C6308/P6308	
---------	------------------------------	--

FEP0CSR — Function Endpoint 0 Control/Status Register**BBH,BAH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	R/C*	R/W	R/C	R/W	R/W	R/C*

.7**Setup Data End Clear Bit**

0	No effect (when write)
1	To clear the Setup Data End Clear bit

.6**Out Packet Ready Clear Bit**

0	No effect (when write)
1	To clear Out Packet Ready Clear bit

.5**Setup Transfer End Bit**

0	No effect (when write)
1	SIE sets this bit when a control transfer ends before Bit 4 is set

.4**Setup Data End Bit**

0	No effect (when write)
1	MCU sets this bit after loading or unloading the last packet data into the FIFO

.3**Stall Signal Receive Bit**

0	MCU clears this bit to end the stall condition
1	SIE sets this bit if a control transaction is ended due to a protocol violation

.2**Stall Signal Sending Bit**

0	No effect (when write)
1	To send stall signal

.1**In Packet Ready Bit**

0	SIE clear this bit once the packet has been successfully sent to the host
1	MCU sets this bit after writing a packet of data into endpoint 0 FIFO

.0**Out Packet Ready Bit**

0	No effect (when write)
1	SIE sets this bit once a valid do ken is written to the FIFO

Note: If FEPSEL register value is zero, this register s configured as above.

FEPINCSR — Function Endpoint In Register**BAH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	—	—	0	0	0
Read/Write	W	W	R	—	—	R/W	R/C	R/W

.7**Data toggle Sequence Clear Bit**

0	No effect (when write)
1	To clear the DATA1/DATA0 toggle bit

.6**FIFO Flush Bit**

0	Cleared by H/W after flush is done
1	To flush FIFO

.5**In Packet Ready Status Bit**

0	It is cleared by the H/W when the data has been transferred to the host
1	When MCU writes a "1" to bit 0, this bit is always set

.4 - .3

Not used for KS86C6308/P6308

.2**Stall Signal Sending Bit**

0	No effect (when write)
1	To send stall signal

.1**FIFO Error Bit**

0	To clear this bit
1	Set by H/W if FIFO under-run error during ISO

.0**In Packet Ready Bit**

0	SIE sets this bit once the packet has been successfully sent to the host
1	MCU sets this bit after writing a packet of data into endpoint IN FIFO

FEPMAXP — Function Endpoint Max Packet Register

BDH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	1	0	0	0	0
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W

.7 - .5

Not used for KS86C6308/P6308

.4 - .0

Max Packet Size

Contains maximum packet size for this Endpoint

Max Packet Size: The maximum size of the buffer for Endpoint and 2 is 16 bytes, if the max packet size is chosen to be 8 or less, two packets may be loaded to buffer. This is recommended for an ISO endpoint, since it needs a constant bandwidth.

FEPOUTCSR — Function Endpoint Out Register**BBH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	—	0	0	0
Read/Write	W	W	R/C	R/C	—	R/W	R/C	R/C

.7**Data Toggle Sequence Clear Bit**

0	No effect (when write)
1	To clear the DATA1/DATA0 toggle bit

.6**FIFO Flush Bit**

0	Cleared by H/W after flush is done
1	To flush FIFO

.5**DATA ERROR Bit**

0	To clear this bit
1	Set by H/W if set Bit 0 even if the core has a CEC/bit stuffing error

.4**Stall Signal Receive Bit**

0	MCU clears this bit to end the stall condition
1	SIE sets this bit if a control transaction is ended due to a protocol violation

.3

Not used for KS86C6308/P6308

.2**Stall Signal Sending Bit**

0	No effect(when write)
1	To send stall signal

.1**FIFO Error Bit**

0	To clear this bit
1	Set by H/W if FIFO under-run error during ISO

.0**Out Packet Ready Bit and Out Packet Ready Clear Bit**

0	No effect (when write)
1	SIE sets this bit once a valid token is written to the FIFO
1	MCU sets to clear Out Packet Ready Bit

FEPSEL — Function Endpoint Selection Register**BEH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	—	—	0	0
Read/Write	—	—	—	—	—	—	R/W	R/W

.7 - .2

Not used for KS86C6308/P6308

.1 - .0**Function Endpoint Selection Bits**

0	0	Selects Endpoint0
0	1	Selects Endpoint1
1	0	Selects Endpoint2

FINTEN — Function Endpoint Interrupt Enable Register**B8H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	—	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

.7 - .3

Not used for KS86C6308/P6308

.2**Endpoint 2 Interrupt Pending Bit**

0	Disable ENDPOINT2 interrupt (default)
1	Enable ENDPOINT2 interrupt

.1**Endpoint 1 Interrupt Pending Bit**

0	Disable ENDPOINT1 interrupt (default)
1	Enable ENDPOINT1 interrupt

.0**Endpoint 0 Interrupt Pending Bit**

0	Disable ENDPOINT0 interrupt (default)
1	Enable ENDPOINT0 interrupt

FINTSTS — Function Endpoint Interrupt Pending Register

B7H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	—	0	0	0
Read/Write	—	—	—	—	—	R/C	R/C	R/C
.7 - .3	Not used for KS86C6308/P6308							
.2	Endpoint 2 Interrupt Status Bit H/W sets this bit upon: Clearing In Packet Ready bit for IN Setting Out Packet Ready bit for OUT							
.1	Endpoint 1 Interrupt Status Bit H/W sets this bit upon: Clearing In Packet Ready bit for IN Setting Out Packet Ready bit for OUT							
.0	Endpoint 0 Interrupt Status Bit H/W sets this bit upon: Setting Out Packet Ready bit Clearing In Packet Ready bit Setting Set End bit Clearing Data End bit							

FLAGS — System Flags Register**D5H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	—	—	—	—
Read/Write	R/W	R/W	R/W	R/W	—	—	—	—

.7

Carry Flag (C)

0	Operation does not generate a carry or borrow condition
---	---

.6

Zero Flag (Z)

0	Operation result is a non-zero value
1	Operation result is zero

.5

Sign Flag (S)

0	Operation generates a positive number (MSB = "0")
1	Operation generates a negative number (MSB = "1")

.4

Overflow Flag (V)

0	Operation result is $\leq +127$ or ≥ -128
1	Operation result is $\geq +127$ or ≤ -128

.3 - .0

Not used for KS86C6308/P6308

FRAMEH — Frame Number High Byte Register

B1H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	0	0	0	0
Read/Write	—	—	—	—	R	R	R	R

.7 - .4	Not used for KS86C6308/P6308
---------	------------------------------

.3	Frame Error Bit
0	No effect
1	If a CRC error is detected, ERR flag is set so that FW can ignore the frame number and use its internal value or wait for the next SOF if this happens on SYNCH_FRAME time.

.2 - .0	Current Frame Number Bits
Upper bits of Frame number that contains the current number received with the last SOF	

FRAMEL — Frame Number Low Byte Register**B0H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

.7 - .0**Current Frame Number Bits**

Lower bits of Frame number that contains the current number received with the last SOF
--

HEP0BCNT — HUB Endpoint 0 Byte Count Register**F8H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	0	0	0	0
Read/Write	—	—	—	—	R	R	R	R

.7 - .4

Not used for KS86C6308/P6308

.3 - .0**Endpoint 0 Byte Count Bits**

The number of valid bytes in Endpoint 0 FIFO
--

HEP0BUF — HUB Endpoint 0 Buffer Register

F9H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 - .0	Endpoint 0 Data Buffer Bits Indicates Endpoint 0 data in 8 byte deep count
---------	--

HEP0CSR — HUB Endpoint 0 Control/Status Register**FAH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	R/C*	R/W	R/C	R/W	R/W	R/C*

.7**Setup Data End Clear Bit**

0	No effect (when write)
1	To clear the Setup Data End Clear bit

.6**Out Packet Ready Clear Bit**

0	No effect (when write)
1	To clear Out Packet Ready Clear bit

.5**Setup Transfer End Bit**

0	No effect (when write)
1	SIE sets this bit when a control transfer ends before Bit 4 is set

.4**Setup Data End Bit**

0	No effect (when write)
1	MCU sets this bit after loading or unloading the last packet data into the FIFO

.3**Stall Signal Receive Bit**

0	MCU clears this bit to end the stall condition
1	SIE sets this bit if a control transaction is ended due to a protocol violation

.2**Stall Signal Sending Bit**

0	No effect (when write)
1	To send stall signal

.1**In Packet Ready Bit**

0	SIE clear this bit once the packet has been successfully sent to the host
1	MCU sets this bit after writing a packet of data into HUB Endpoint 0 FIFO

.0**Out Packet Ready Bit**

0	No effect (when write)
1	SIE sets this bit once a valid token is written to the FIFO

HINTEN — HUB Interrupt Enable Register**EBH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	1	0	—	—	—	—
Read/Write	R/W	R/W	R/W	R/W	—	—	—	—

.7

HUB Interrupt Enable Bit	
0	Interrupt disable
1	Interrupt enable

.6

Suspend and Resume Interrupt Enable Bit	
0	Interrupt disable
1	Interrupt enable

.5

Connection Interrupt Enable Bit	
0	Interrupt disable
1	Interrupt enable

.4

Function Interrupt Enable Bit	
0	Interrupt disable
1	Interrupt enable

.3 - .0

Not used for KS86C6308/P6308

HINTSTS — HUB Interrupt Pending Register**EAH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	–	–	–	–
Read/Write	R/C	R/C	R/C	R/C	–	–	–	–

.7**HUB Interrupt Pending Bit**

0	No effect
1	Set if HUB Endpoint 0 is serviced
1	Write 1 to clear pending bit

.6**Suspend and Resume Interrupt Pending Bit**

0	No effect
1	Set if Suspend or Resume signal is received
1	Write 1 to clear pending bit

.5**Connection Interrupt Pending Bit**

0	No effect
1	Set if HUB port is connected to function
1	Write 1 to clear pending bit

.4**Function Interrupt Pending Bit**

0	No effect
1	This bit is set when a bit in the Function Status Change register has 1 in it
1	Write 1 to clear pending bit

.3 - .0

Not used for KS86C6308/P6308

INTMASK — Interrupt Mask Register**B3H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	—	0	—	—
Read/Write	—	—	—	—	—	R/W	—	—

.7 - .3

Not used for KS86C6308/P6308

.2**USB Start of Frame Interrupt (USOFINT) Enable Bit**

0	Interrupt disable
1	Interrupt enable

.1 - .0

Not used for KS86C6308/P6308

INTPEND — Interrupt Pending Register

B2H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	—	0	—	—
Read/Write	—	—	—	—	—	R/W	—	—
.7 - .3	Not used for KS86C6308/P6308							
.2	USB Start of Frame Interrupt (USOFINT) Pending Bit							
	0	No pending (when read)/clear pending bit (when write)						
	1	Pending (when read)/no effect (when write)						
.1 - .0	Not used for KS86C6308/P6308							

MISCCON — Miscellaneous Control Register**E3H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	—	—	0	0
Read/Write	—	—	—	—	—	—	R/W	R/W

.7 - .2

Not mapped for KS86C6308

.1**MCU Handling of HUB Commands Enable Bit**

0	Command decoder will not take exception, instead will execute the request or return the STALL to the host.
1	Command decoder will take exception, whenever GET_DESCRIPTOR or Unknown request from the Host is decoded

.0**USB RESET Enable Bit**

0	USB_RSTN remains high
1	USB_RSTN is asserted when USB reset is detected.

PORTSEL — Port Selection Register**FBH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	—	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

.7 - .3

Not used for KS86C6308/P6308

.2 - .0**Port Selection for Status Transfer Bits**

0	0	1	Selects downstream Port1
0	1	0	Selects downstream Port2
0	1	1	Selects downstream Port3
1	0	0	Selects downstream Port4
1	0	1	Selects embedded port

USRECON — Suspend/Resume Control Register**E4H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	—	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

.7 - .3

Not used for KS86C6308/P6308

.2**HUB Suspend In Status Bit**

0	No effect
1	SUSPEND_OUT signal is generated by the MCU when HUB enters low power suspend mode (2ms after HUB Suspend has been set).

.1**HUB Resume In Status Bit**

0	No effect
1	RESUME_IN set by the MCU to initiate remote resume

.0**HUB Suspend Status Bit**

0	No effect
1	This bit is set when the HUB controller detects no activity on the root HUB in excess of 3ms and the HUB is ready to go into low power suspend mode.

SYM — System Mode Register**DFH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	—	—	0	0	0	0
Read/Write	—	—	—	—	R/W	R/W	R/W	R/W

.7-.4

Not used for KS86C6308/P6308

.3**Global Interrupt Enable Bit ^(note)**

0	Disable global interrupt processing
1	Enable global interrupt processing

.2 - .0**Page Selection Bits**

0	0	0	Addressing page 0 locations for KS86C6308/P6308
Other values			Enable global interrupt processing

NOTE: SYM must be selected bit 2, bit 1, and 0 into 000 for KS86C6308/P6308.

HEP1STS0 — HUB Endpoint 1 Status Low Byte Register**F0H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

.7 - .1

Reserved. These bits return 0 when read.

.0**INT0_Stall**

0	If the HUB endpoint 1 is not halted, then halt feature is reset to zero
1	If the HUB endpoint 1 is halted, then halt feature is set to one

HEP0STS0 — HUB Endpoint 0 Status Low Byte Register**F2H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

.7 - .1

Reserved. These bits return 0 when read.

.0**INT0_Stall**

0	If the HUB endpoint 0 is not halted, then halt feature is reset to zero
1	If the HUB endpoint 0 is halted, then halt feature is set to one

HSTS0 — HUB Status Low Byte Register**F4H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

.7 - .1

Reserved. These bits return 0 when read.

.0**Over-current Indicator**

0	No over-current condition currently exists
1	A HUB over-current condition exists

.0**Local Power Source**

0	Local power supply good
1	Local power supply lost (inactive)

HSTS2 — HUB Status Change Low Byte Register**F6H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

.7 - .1

Reserved. These bits return 0 when read.

.0**Over-Current Indicator Change**

0	No change has occurred to Local Power Status
1	Over-Current indicator has changed

.0**Local Power Status Change**

0	No change has occurred to Over-Current Indicator
1	Over-Current Indicator has changed

PXSTS0 — Port Status Low Byte Register**FCH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

.7 - .5

Reserved. These bits return 0 when read.

.4**PORT_RESET**

0	Reset signaling not asserted
1	Reset signaling asserted

.3**PORT_OVER_CURRENT**

0	All no over-current condition exists on this port
1	An over-current condition exists on this port

.2**PORT_SUSPEND**

0	Not suspended
1	Suspended or resuming

.1**PORT_ENABLE**

0	Port is disabled
1	Port is enabled

.0**PORT_CONNECTION**

0	No device is present
1	A devices is present on this port

PXSTS1 — Port Status High Byte Register

FDH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

.7 - .2

Reserved. These bits return 0 when read.

.1	PORT_LOW_SPEED							
	0	Full-speed device attached to this port						
	1	Low-speed device attached to this port						

.0	PORT_POWER							
	0	This port is in the Powered-off state						
	1	This port is not in the Powered-off state						

PXSTS2 — Port Status Change Low Byte Register**FEH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

.7 - .5 Reserved. These bits return 0 when read.

.4 **C_PORT_RESET**

0	No change
1	Reset complete

.3 **C_PORT_OVER_CURRENT**

0	No change has occurred to Over-Current Indicator
1	Over-Current Indicator has changed

.2 **PORT_SUSPEND**

0	Not change
1	Resume complete

.1 **C_PORT_ENABLE**

0	Not change
1	Current Port status has changed

.0 **C_PORT_CONNECTION**

0	No change has occurred to Current Connect status
1	Current Connect status has changed

PXSTS3 — Port Status Change High Byte Register**FFH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

.7 - .1 Reserved. These bits return 0 when read.

P0CONH — Port 0 Control Register (High Byte)**A1H****Bit Identifier****RESET Value****Read/Write**

.7	.6	.5	.4	.3	.2	.1	.0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6**P0.7 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edge external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.5 and .4**P0.6 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edge external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.3 and .2**P0.5 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edge external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.1 and .0**P0.4 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edge external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

P0CONL — Port 0 Control Register (Low Byte)**A0H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6**P0.3 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edge external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.5 and .4**P0.2 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edge external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.3 and .2**P0.1 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edge external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.1 and .0**P0.0 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edge external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

POINT — Port 0 Interrupt Control Register**A6H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7**P0.7 Configuration Bits**

0	External interrupt disable
1	External interrupt enable

.6**P0.6 Configuration Bits**

0	External interrupt disable
1	External interrupt enable

.5**P0.5 Configuration Bits**

0	External interrupt disable
1	External interrupt enable

.4**P0.4 Configuration Bits**

0	External interrupt disable
1	External interrupt enable

.3**P0.3 Configuration Bits**

0	External interrupt disable
1	External interrupt enable

.2**P0.2 Configuration Bits**

0	External interrupt disable
1	External interrupt enable

.1**P0.1 Configuration Bits**

0	External interrupt disable
1	External interrupt enable

.0**P0.0 Configuration Bits**

0	External interrupt disable
1	External interrupt enable

P0PND — Port 0 Interrupt Pending Register**A7H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (NOTE)	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7**P0.7 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.6**P0.6 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.5**P0.5 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.4**P0.4 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.3**P0.3 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.2**P0.2 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.1**P0.1 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.0**P0.0 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

P1CONH — Port 1 Control Register (High Byte)**A3H****Bit Identifier****RESET Value****Read/Write**

.7	.6	.5	.4	.3	.2	.1	.0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6**P1.7 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.5 and .4**P1.6 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.3 and .2**P1.5 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.1 and .0**P1.4 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

P1CONL — Port 1 Control Register (Low Byte)**A4H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6**P1.3 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.5 and .4**P1.2 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.3 and .2**P1.1 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.1 and .0**P1.0 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

P2CONH — Port 2 Control Register (High Byte)**A5H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6**Port2,P2.7 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edges external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.5 and .4**P2.6 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edges external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.3 and .2**P2.5 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edges external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.1 and .0**P2.4 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edges external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

P2CONL — Port 2 Control Register (Low Byte)**A4H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6**Port 2, P2.3 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edges external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.5 and .4**Port 2, P2.2 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edges external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.3 and .2**Port 2, P2.1 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edges external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.1 and .0**Port 2, P2.0 Configuration Bits**

0	0	Schmitt trigger input, rising edge external interrupt
0	1	Schmitt trigger input, falling edges external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

P2INT — Port 2 Interrupt Enable Register**A8H****Bit Identifier****RESET Value****Read/Write**

.7	.6	.5	.4	.3	.2	.1	.0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7**P2.7 Interrupt Enable Bit**

0	External interrupt disable
1	External interrupt enable

.6**P2.6 Interrupt Enable Bit**

0	External interrupt disable
1	External interrupt enable

.5**P2.5 Interrupt Enable Bit**

0	External interrupt disable
1	External interrupt enable

.4**P2.4 Interrupt Enable Bit**

0	External interrupt disable
1	External interrupt enable

.3**P2.3 Interrupt Enable Bit**

0	External interrupt disable
1	External interrupt enable

.2**P2.2 Interrupt Enable Bit**

0	External interrupt disable
1	External interrupt enable

.1**P2.1 Interrupt Enable Bit**

0	External interrupt disable
1	External interrupt enable

.0**P2.0 Interrupt Enable Bit**

0	External interrupt disable
1	External interrupt enable

P2PND — Port 2 Interrupt Pending Register**A9H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (NOTE)	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7**P2.7 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.6**P2.6 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.5**P2.5 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.4**P2.4 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.3**P2.3 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.2**P2.2 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.1**P2.1 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.0**P2.0 Interrupt Pending Bit**

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

NOTE: To clear a port 2 interrupt pending condition, write a "0" to the corresponding P2PND register bit location.

P3CON — Port 3 Control Register

AAH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6

Port 3, P3.3 Configuration Bits

0	0	Schmitt trigger input, Timer A external clock input(TACLK)
0	1	System clock output(CLO) mode. CLO comes from system clock circuit.
1	0	Push-pull output
1	1	N-channel open-drain output mode

.5 and .4

Port 3, P3.2 Configuration Bits

0	0	Schmitt trigger input, Timer1 external clock input(TBCLK)
0	1	USB clock output(USBCLK) comes from PLL
1	0	Push-pull output
1	1	N-channel open-drain output mode

.3 and .2

Port 3, P3.1 Configuration Bits

0	0	Schmitt trigger input, Timer1 capture input(TBCAP)
0	1	Timer0 PWM or Match output(TAOUT)
1	0	Push-pull output
1	1	N-channel open-drain output mode

.1 and .0

Port 3, P3.0 Configuration Bits

0	0	Schmitt trigger input, Timer0 capture input(TACAP)
0	1	Timer PWM or Match output(TAOUT)
1	0	Push-pull output
1	1	N-channel open-drain output mode

P4CON — Port 4 Control Register**ABH****Bit Identifier****RESET Value****Read/Write**

.7	.6	.5	.4	.3	.2	.1	.0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7-.4.**Not mapped for KS86C6308/P6308****.3 and .2****Port 4, P4.1 Configuration Control Bits**

0	0	Schmitt trigger input, falling edge external interrupt with pull-up
0	1	N-CH open drain output mode with pull-up
1	0	N-CH open drain output mode
1	1	Output pull-pull mode

.1 and .0**Port 4, P4.0 Configuration Control Bits**

0	0	Schmitt trigger input, falling edge external interrupt with pull-up
0	1	N-CH open drain output mode with pull-up
1	0	N-CH open drain output mode
1	1	Output pull-pull mode

P4INTPND — Port 4 Interrupt Enable and Pending Register**ACH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7-.4**Not mapped for KS86C6308/P6308****.3****P4.1 Interrupt enable Bit**

0	External interrupt disable
1	External interrupt enable

.2**P4.0 Interrupt enable Bit**

0	External interrupt disable
1	External interrupt enable

.1**P4.1 Interrupt Pending Bit**

0	No pending (when bit is read)/clear pending bit (when bit is write)
1	Pending (when bit is read)/no effect (when bit is write)

.0**P4.0 Interrupt Pending Bit**

0	No pending (when bit is read)/clear pending bit (when bit is write)
1	Pending (when bit is read)/no effect (when bit is write)

GPIOCONINT — GPIO Control and interrupt pending Register**99H****(GPIO Mode only)**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6**DP0/GPIO Configuration Control Bits**

0	0	Schmitt trigger input, falling edge external interrupt
0	1	Schmitt trigger input, falling edge external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.5 and .4**DM0/GPIO Configuration Control Bits**

0	0	Schmitt trigger input, falling edge external interrupt
0	1	Schmitt trigger input, falling edge external interrupt with pull-up
1	0	N-CH open drain output mode
1	1	N-CH open drain output mode with pull-up

.3**DP0/GPIO Interrupt Enable Bit**

0	External interrupt disable
1	External interrupt enable

.2**DM0/GPIO Interrupt Enable Bit**

0	External interrupt disable
1	External interrupt enable

.1**DP0/GPIO Interrupt Pending Bit**

0	No pending (when bit is read)/clear pending bit (when bit is write)
1	Pending (when bit is read)/no effect (when bit is write)

.0**DM0/GPIO Interrupt Pending Bit**

0	No pending (when bit is read)/clear pending bit (when bit is write)
1	Pending (when bit is read)/no effect (when bit is write)

TACON — Timer A Control Register**D2H**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6**TA Counter Input Clock Selection Bits**

0	0	CPU clock/1024
0	1	CPU clock/256
1	0	CPU clock/64
1	1	External clock(TACLK)

.5 and .4**TA Operating Mode Selection Bits**

0	0	Interval timer mode (TAOUT)
0	1	Capture mode(capture on rising edge, counter running, OVF can occur)
1	0	Capture mode(capture on falling edge, counter running, OVF can occur)
1	1	PWM mode (OVF interrupt can occur)

.3**TA Counter Clear Bit (TACLR)**

0	No effect when written
1	Clear Timer A counter

.2**TA Overflow Interrupt Enable Bit (TAOVF)**

0	Disable TA overflow interrupt
1	Enable TA overflow interrupt

.1**TA Match/Capture Interrupt Enable Bit (TAINT)**

0	Disable TA match interrupt
1	Enable TA match interrupt

.0**TA Match/Capture Interrupt Pending Bit (TAPND)**

0	No interrupt pending, Clear pending bit(write) Interrupt is pending
1	Interrupt is pending(when read)/No effect(when write)

TBCON — Timer B Control Register**9EH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6 and .5**TB Counter Input Clock Selection Bits**

0	0	0	fx/1024
0	1	0	fx/256
1	0	0	fx/64
1	1	0	fx/8
0	1	1	fx/1
0	1	1	External clock(TBCLK) Falling edge
1	0	1	External clock(TBCLK) Falling edge
1	1	1	Counter stop

.4 and .3**TB Operating Mode Selection Bits**

0	0	Interval timer mode (TBOU)
0	1	Capture mode(capture on rising edge, counter running, OVF can occur)
1	0	Capture mode(capture on falling edge, counter running, OVF can occur)
1	1	PWM mode (OVF & match interrupt can occur)

.2**TB Counter Clear Bit**

0	No effect
1	Clear Timer1 counter(when write)

.1**TB Match/Capture Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.0**TB Overflow Interrupt Enable Bit**

0	Disable overflow interrupt
1	Enable overflow interrupt

USBSEL — USB/GPIO Mode select Register**FBH**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	0	0	0	0	0	0
Read/Write	–	–	R/W	R/W	R/W	R/W	R/W	R/W

.7 - .6

Not used for KS86C6308/P6308

.5**Upstream Mode select Bit**

0	Upstream port pull-up resistor disable(GPIO mode)
1	Upstream port pull-up register enable

.4**Downstream Mode select Bit**

0	Downstream port4 disable
1	Downstream port4 enable

.3**Downstream Mode select Bit**

0	Downstream port3 disable
1	Downstream port3 enable

.2**Downstream Mode select Bit**

0	Downstream port2 disable
1	Downstream port2 enable

.1**Downstream Mode select Bit**

0	Downstream port1 disable
1	Downstream port1 enable

.0**USB/GPIO Mode select Bit**

0	GPIO mode(GPIO port)
1	USB global mode(USB port)

USBMODE — USB Mode select Register

96H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	–	–	0	0	0
Read/Write	–	–	–	–	–	R/W	R/W	R/W

.7 - .3	Not used for KS86C6308/P6308							
---------	------------------------------	--	--	--	--	--	--	--

.2	Self Power							
	0	Disable self power						
	1	Enable self power						

.1	RESUME_IN							
	0	Disable remote RESUME_IN						
	1	Enable remote RESUME_IN						

.2	USB Function							
	0	Disable USB function						
	1	Enable USB function						

FSXCON — Full Speed Tranceiver Crossover Point Control Register

E8H

(This register will be used to control the USB Tranceiver signal quality)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	0	0	0	0	0	0
Read/Write	-	-	R/W	R/W	R/W	R/W	R/W	R/W
.7-.6	Not used for KS86C6308/P6308							
.5-.0	USB Signal Crossover Point Control Bit							

LSXCON — Low Speed Transceiver Crossover Point Control Register

E9H

(This register will be used to control the USB Transceiver signal quality)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	—	—	0	0	0	0	0	0
Read/Write	-	-	R/W	R/W	R/W	R/W	R/W	R/W

.7-.6

Not used for KS86C6308/P6308

.5-.0

USB Signal Crossover Point Control Bit

Edge delay Control	Bit 5, (2)	Bit 4, (1)	Bit 3, (0)	Delay Value	Delay Unit
RISE edge	0	0	0	0	(about) 2.5nsec
		0	1	1	
		1	0	2	
		1	1	4	
FALL edge	1	0	0	0	
		0	1	1	
		1	0	2	
		1	1	4	

5

INTERRUPT STRUCTURE

OVERVIEW

The SAM88RCRI interrupt structure has two basic components: a vector, and sources. The number of interrupt sources can be serviced through a interrupt vector which is assigned in ROM address 0000H-0001H.

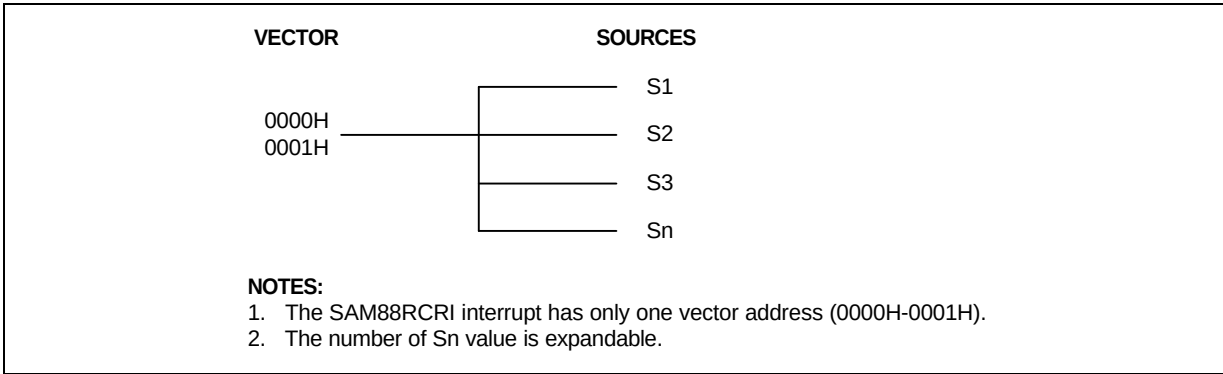


Figure 5-1. KS86-Series Interrupt Type

INTERRUPT PROCESSING CONTROL POINTS

Interrupt processing can be controlled in two ways: either globally, or by specific interrupt level and source. The system-level control points in the interrupt structure are therefore:

- Global interrupt enable and disable (by EI and DI instructions)
- Interrupt source enable and disable settings in the corresponding peripheral control register(s)

ENABLE/DISABLE INTERRUPT INSTRUCTIONS (EI, DI)

The system mode register, SYM (DFH), is used to enable and disable interrupt processing.

SYM.3 is the enable and disable bit for global interrupt processing respectively, by modifying SYM.3. An Enable Interrupt (EI) instruction must be included in the initialization routine that follows a reset operation in order to enable interrupt processing. Although you can manipulate SYM.3 directly to enable and disable interrupts during normal operation, we recommend that you use the EI and DI instructions for this purpose.

INTERRUPT PENDING FUNCTION TYPES

When the interrupt service routine has executed, the application program's service routine must clear the appropriate pending bit before the return from interrupt subroutine (IRET) occurs.

INTERRUPT PRIORITY

Because there is not a interrupt priority register in SAM88RCRI, the order of service is determined by a sequence of source which is executed in interrupt service routine.

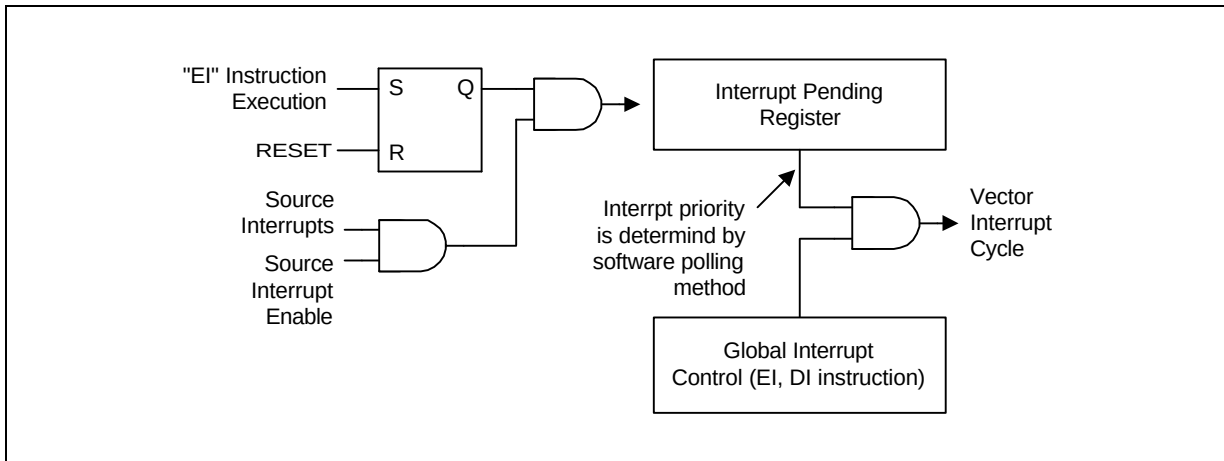


Figure 5-2. Interrupt Function Diagram

INTERRUPT SOURCE SERVICE SEQUENCE

The interrupt request polling and servicing sequence is as follows:

1. A source generates an interrupt request by setting the interrupt request pending bit to "1".
2. The CPU generates an interrupt acknowledge signal.
3. The service routine starts and the source's pending flag is cleared to "0" by software.
4. Interrupt priority must be determined by software polling method.

INTERRUPT SERVICE ROUTINES

Before an interrupt request can be serviced, the following conditions must be met:

- Interrupt processing must be enabled (EI, SYM.3 = "1")
- Interrupt must be enabled at the interrupt's source (peripheral control register)

If all of the above conditions are met, the interrupt request is acknowledged at the end of the instruction cycle. The CPU then initiates an interrupt machine cycle that completes the following processing sequence:

1. Reset (clear to "0") the global interrupt enable bit in the SYM register (DI, SYM.3 = "0") to disable all subsequent interrupts.
2. Save the program counter and status flags to stack.
3. Branch to the interrupt vector to fetch the service routine's address.
4. Pass control to the interrupt service routine.

When the interrupt service routine is completed, an Interrupt Return instruction (IRET) occurs. The IRET restores the PC and status flags and sets SYM.3 to "1"(EI), allowing the CPU to process the next interrupt request.

GENERATING INTERRUPT VECTOR ADDRESSES

The interrupt vector area in the ROM contains the address of the interrupt service routine. Vectored interrupt processing follows this sequence:

1. Push the program counter's low-byte value to stack.
2. Push the program counter's high-byte value to stack.
3. Push the FLAGS register values to stack.
4. Fetch the service routine's high-byte address from the vector address 0000H.
5. Fetch the service routine's low-byte address from the vector address 0001H.
6. Branch to the service routine specified by the 16-bit vector address.

KS86C6308/P6308 INTERRUPT STRUCTURE

The KS86C6308/P6308 microcontroller has 27 peripheral interrupt sources:

- Timer A match/capture/overflow interrupt
- Timer B match/capture/overflow interrupt
- Eight external interrupts for port 2, P2.0-P2.7
- Two external interrupts for port 4, P4.0-P4.1
- USB interrupt and GPIO interrupt
- USB SOF interrupt
- USB function interrupt
- USB HUB interrupt

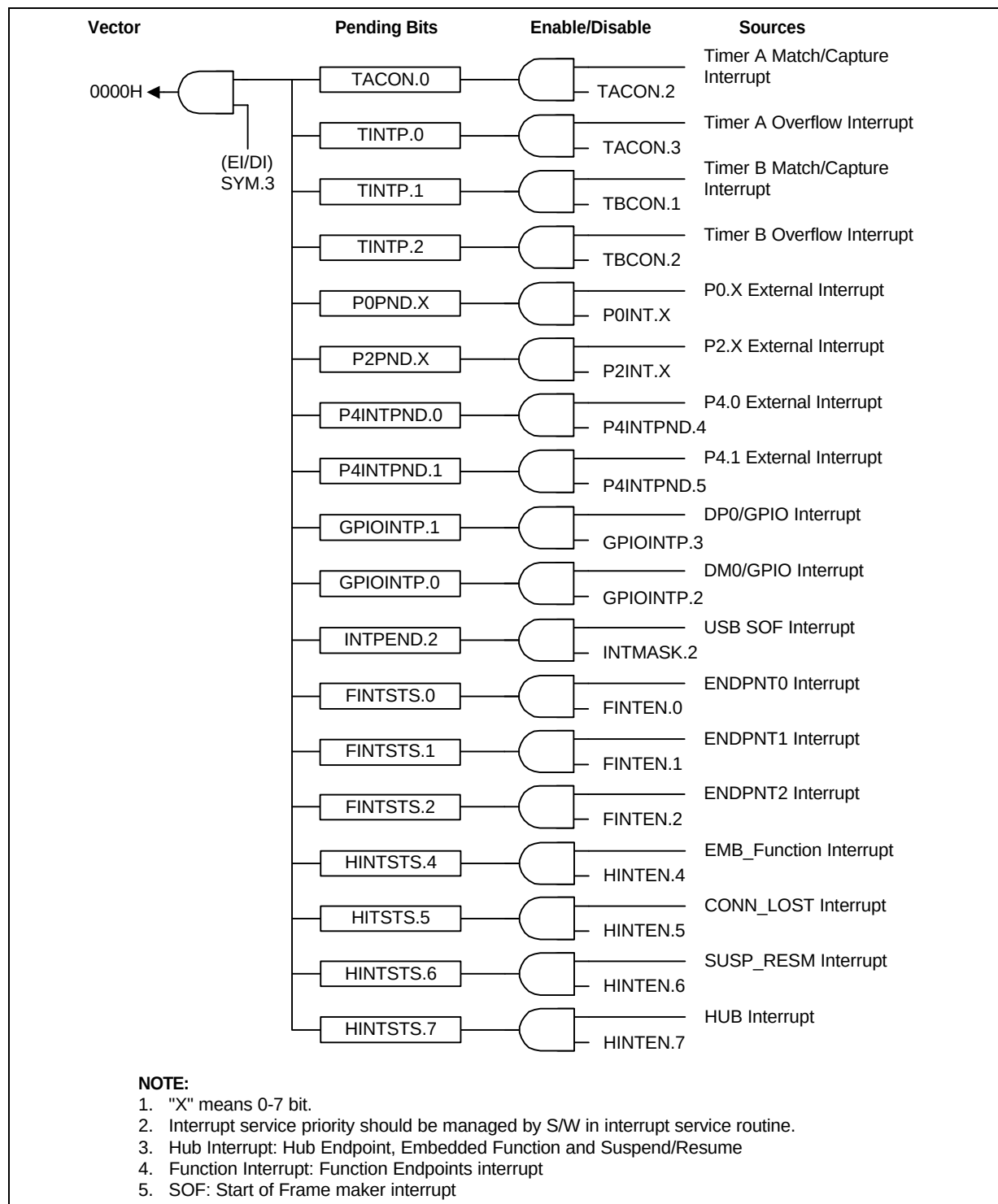


Figure 5-3. KS86C6308/P6308 Interrupt Structure

NOTES

Clock Circuit

RESET and Power-Down

I/O Ports

Basic Timer and Timer A, Timer B

Universal Serial Bus

Electrical Data

Mechanical Data

KS86P6308 OTP

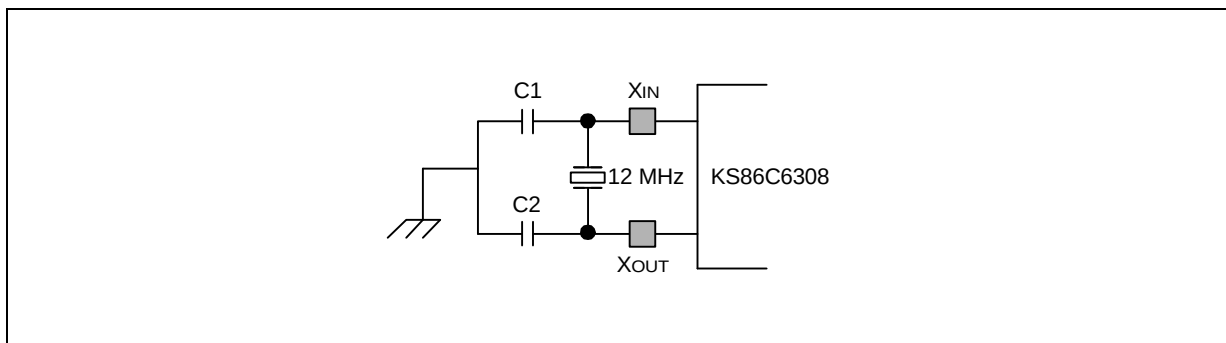
Development Tools

7

CLOCK CIRCUIT

OVERVIEW

An external crystal or ceramic oscillation source provides only 12 MHz clock for the KS86C6308. The X_{IN} and X_{OUT} pins connect the oscillation source to the on-chip clock circuit. External clock and crystal/ceramic oscillator circuits are shown in Figures 7-1.



**Figure 7-1. Main Oscillator Circuit
(External Crystal/Ceramic Oscillator)**

MAIN OSCILLATOR LOGIC

To increase processing speed and to reduce clock noise, non-divided logic is implemented for the main oscillator circuit. For this reason, very high resolution waveforms (square signal edges) must be generated in order for the CPU to efficiently process logic operations.

CLOCK STATUS DURING POWER-DOWN MODES

The two power-down modes, Stop mode and Idle mode, affect clock oscillation as follows:

- In Stop mode, the main oscillator "freezes," halting the CPU and peripherals. The contents of the register file and current system register values are retained. Stop mode is released, and the oscillator started, by a reset operation or by an external interrupt with RC-delay noise filter (for KS86C6308).
- In Idle mode, the internal clock signal is gated off to the CPU, but not to interrupt control and the timer. The current CPU status is preserved, including stack pointer, program counter, and flags. Data in the register file is retained. Idle mode is released by a reset or by an interrupt (external or internally-generated).

SYSTEM CLOCK CONTROL REGISTER (CLKCON)

The system clock control register, CLKCON, is located in location D4H. It is read/write addressable and has the following functions:

- Oscillator IRQ wake-up function enable/disable (CLKCON.7)
- Oscillator frequency divide-by value: non-divided, 2, 8, or 16 (CLKCON.4 and CLKCON.3)

The CLKCON register controls whether or not an external interrupt can be used to trigger a Stop mode release (This is called the "IRQ wake-up" function). The IRQ wake-up enable bit is CLKCON.7.

After a reset, the external interrupt oscillator wake-up function is enabled, the main oscillator is activated, and the $f_{OSC}/16$ (the slowest clock speed) is selected as the CPU clock. If necessary, you can then increase the CPU clock speed to f_{OSC} , $f_{OSC}/2$ or $f_{OSC}/8$.

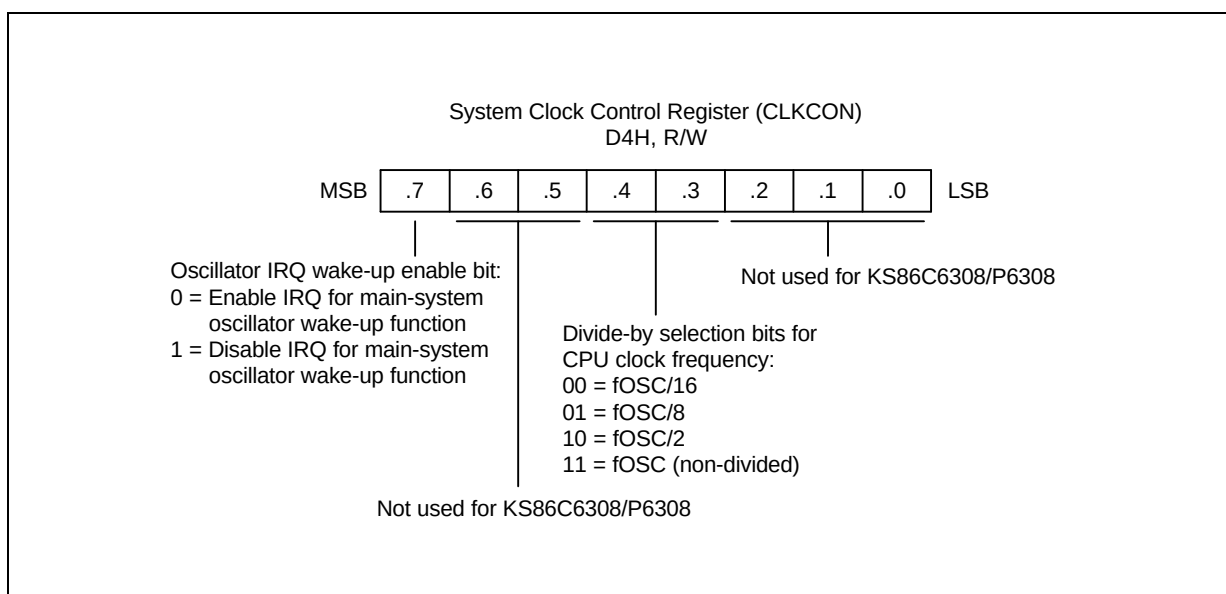


Figure 7-2. System Clock Control Register (CLKCON)

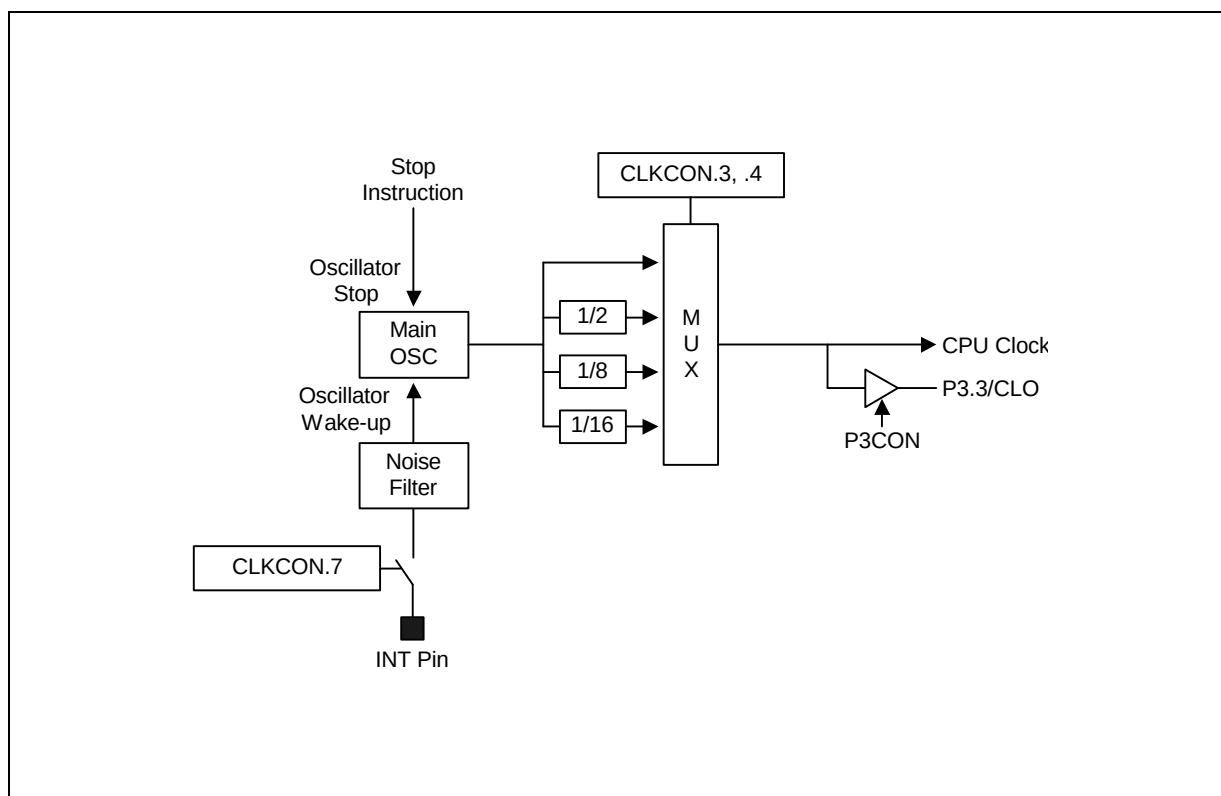


Figure 7-3. System Clock Circuit Diagram

NOTES

8

RESET and POWER-DOWN

SYSTEM RESET

OVERVIEW

During a power-on reset, the voltage at V_{DD} is High level and the RESET pin is forced to Low level. The RESET signal is input through a Schmitt trigger circuit where it is then synchronized with the CPU clock. This brings the KS86C6308 into a known operating status.

The RESET pin must be held to Low level for a minimum time interval after the power supply comes within tolerance in order to allow time for internal CPU clock oscillation to stabilize. The minimum required oscillation stabilization time for a reset is approximately 5.5ms ($= 2^{16}/f_{OSC}$, $f_{OSC} = 12$ MHz).

When a reset occurs during normal operation (with both V_{DD} and RESET at High level), the signal at the RESET pin is forced Low and the reset operation starts. All system and peripheral control registers are then set to their default hardware reset values (see Table 8-1).

The following sequence of events occur during a reset operation:

- All interrupts are disabled.
- The watchdog function (basic timer) is enabled.
- Ports 0 and 1 are set to input mode.
- Peripheral control and data registers are disabled and reset to their initial values.
- The program counter is loaded with the ROM reset address, 0100H.
- When the programmed oscillation stabilization time interval has elapsed, the address stored in ROM location 0100H (and 0101H) is fetched and executed.

NOTE

To program the duration of the oscillation stabilization interval, you must make the appropriate settings to the basic timer control register, BTCON, before entering Stop mode. Also, if you do not want to use the basic timer watchdog function (which causes a system reset if a basic timer counter overflow occurs), you can disable it by writing '1010B' to the upper nibble of BTCON.

POWER-DOWN MODES

STOP MODE

Stop mode is invoked by the instruction STOP (OPCODE 7FH). In Stop mode, the operation of the CPU and all peripherals is halted. That is, the on-chip main oscillator stops and the supply current is reduced to less than 300 μ A. All system functions are halted when the clock "freezes," but data stored in the internal register file is retained. Stop mode can be released in one of two ways: by a RESET signal or by an external interrupt.

Using RESET to Release Stop Mode

Stop mode is released when the RESET signal is released and returns to High level. All system and peripheral control registers are then reset to their default values and the contents of all data registers are retained. RESET operation automatically selects a slow clock (1/16) because CLKCON.3 and CLKCON.4 are cleared to '00B'. After the oscillation stabilization interval has elapsed, the CPU executes the system initialization routine by fetching the 16-bit address stored in ROM locations 0100H and 0101H.

Using an External Interrupt to Release Stop Mode

Only external interrupts with an RC-delay noise filter circuit can be used to release Stop mode (Clock-related external interrupts cannot be used). External interrupts in the KS86C6308 interrupt structure meet this criteria.

Note that when Stop mode is released by an external interrupt, the current values in system and peripheral control registers are not changed. When you use an interrupt to release Stop mode, the CLKCON.3 and CLKCON.4 register values remain unchanged, and the currently selected clock value is used. If you use an external interrupt for Stop mode release, you can also program the duration of the oscillation stabilization interval. To do this, you must make the appropriate control and clock settings *before* entering Stop mode.

The external interrupt is serviced when the Stop mode release occurs. Following the IRET from the service routine, the instruction immediately following the one that initiated Stop mode is executed.

NOTE

Do not use the STOP mode if you are using an external clock source because X_{IN} input must be restricted internally to V_{SS} to reduce current leakage.

IDLE MODE

Idle mode is invoked by the instruction IDLE (opcode 6FH). In Idle mode, CPU operations are halted while selected peripherals remain active. During Idle mode, the internal clock signal is gated off to the CPU, but not to interrupt logic and timer/counters. Port pins retain the mode (input or output) they had at the time Idle mode was entered.

There are two ways to release Idle mode:

1. Execute RESET. All system and peripheral control registers are reset to their default values and the contents of all data registers are retained. The reset automatically selects a slow clock (1/16) because CLKCON.3 and CLKCON.4 are cleared to '00B'. If interrupts are masked, RESET is the only way to release Idle mode.
2. Activate any enabled interrupt, causing Idle mode to be released. When you use an interrupt to release Idle mode, the CLKCON.3 and CLKCON.4 register values remain unchanged, and the currently selected clock value is used. The interrupt is then serviced. Following the IRET from the service routine, the instruction immediately following the one that initiated Idle mode is executed.

NOTE

Only external interrupts that are not clock-related can be used to release Stop mode. To release Idle mode, however, any type of interrupt (that is, internal or external) can be used.

HARDWARE RESET VALUES

Table 8-1 list the reset values for CPU and system registers, peripheral control registers, and peripheral data registers following a reset operation in normal operating mode. The following notation is used in these tables to represent specific reset values:

- A "1" or a "0" shows the RESET bit value as logic one or logic zero, respectively.
- An 'x' means that the bit value is undefined following RESET.
- A dash ('-') means that the bit is either not used or not mapped.

Table 8-1. Register Values After RESET

Register Function	Register Name	Address	R/W	Bit Values After RESET							
				7	6	5	4	3	2	1	0
General Purpose Register File & Stack Area		00-8F	R/W	x	x	x	x	x	x	x	x
Working Register Area		C0-CF	R/W	x	x	x	x	x	x	x	x
Timer A Counter	TACNT	D0	R	0	0	0	0	0	0	0	0
Timer A Data	TADATA	D1	R/W	1	1	1	1	1	1	1	1
Timer A Control	TACON	D2	R/W	0	0	0	0	0	0	0	0
Clock Control	CLKCON	D4	R/W	0	–	–	0	0	–	–	–
System Flags	FLAGS	D5	R/W	0	0	0	0	–	–	–	–
Stack Pointer	SP	D9	R/W	x	x	x	x	x	x	x	x
Location DB is not mapped.											
Basic Timer Control	BTCON	DC	R/W	0	0	0	0	0	0	0	0
Basic Timer Counter	BTCNT	DD	R	0	0	0	0	0	0	0	0
Location DE is not mapped.											
System Mode	SYM	DF	R/W	–	–	–	–	0	0	0	0
Port 0 Data	P0	E0	R/W	0	0	0	0	0	0	0	0
Port 1 Data	P1	E1	R/W	0	0	0	0	0	0	0	0
Status Change Endpoint	STSCHGEP	E2	R/W	0	0	0	0	0	0	0	0
Miscellaneous Control	MISCCON	E3	R/W	–	–	–	–	0	0	0	0
Suspend/Resume Control	SUSRECON	E4	R/W	–	–	–	–	0	0	0	0
Port Bus Status Port Selected by PORTSEL Bit = VMIN, Bit = VPIN Bit7 to Bit2 = 0	PBUSSTE	E5	R/W	0	0	0	0	0	0	0	0

Table 8-1. Register Values After RESET (continued)

Register Function	Register Name	Address	R/W	Bit Values After RESET							
				7	6	5	4	3	2	1	0
HUB Interface Status LSB	HINTFSTS0	E6	R/W	0	0	0	0	0	0	0	0
HUB Interface Status MSB	HINTFSTS1	E7	R/W	0	0	0	0	0	0	0	0
Full Speed Transceiver Crossover Point Control	FSXCON	E8	R/W	–	–	0	0	0	0	0	0
Low Speed Transceiver Crossover Point Control	LSXCON	E9	R/W	–	–	0	0	0	0	0	0
HUB Interrupt Status	HINTSTS	EA	R/W	0	0	0	0	0	0	0	0
HUB Interrupt Enable	HINTEN	EB	R/W	0	0	0	0	0	0	0	0
Embedded Function Change Status Register. Similar to embedded port register. Refer to explanation above.	EMBFCSTS	EC	R/W	0	0	0	0	0	0	0	0
Embedded Function Status Register, it is similar to embedded port status register but this one is for MCU, while embedded port register is for the Host.	EMBFSTS	ED	R/W	0	0	0	0	0	0	0	0
Device Status LSB	DSTS0	EE	R/W	0	0	0	0	0	0	0	0
Device Status MSB	DSTS1	EF	R/W	0	0	0	0	0	0	0	0
HUB Endpoint 1 Status LSB	HEP1STS0	F0	R	0	0	0	0	0	0	0	0
HUB Endpoint 1 Status MSB	HEP1STS1	F1	R	0	0	0	0	0	0	0	0
HUB Endpoint 0 Status LSB	HEP0STS0	F2	R	0	0	0	0	0	0	0	0
HUB Endpoint 0 Status MSB	HEP0STS1	F3	R	0	0	0	0	0	0	0	0
HUB Status Register LSB	HSTS0	F4	R	0	0	0	0	0	0	0	0
HUB Status Register MSB	HSTS1	F5	R	0	0	0	0	0	0	0	0
HUB Change Status Register LSB	HSTS2	F6	R	0	0	0	0	0	0	0	0
HUB Change Status Register MSB	HSTS3	F7	R	0	0	0	0	0	0	0	0
HUB Endpoint Byte count register, indicates bytes present in the OUT FIFO	HEP0BCNT	F8	R	0	0	0	0	0	0	0	0
HUB Endpoint 0 data buffer	HEP0BUF	F9	R/W	0	0	0	0	0	0	0	0
HUB Endpoint 0 control/status	HEP0CSR	FA	R/W	0	0	0	0	0	0	0	0
Port Select, selects port for status transfer = 1 downstream port 1 = 2 downstream port 2 = 3 downstream port 2 = 4 downstream port 4 = 5 embedded port	PORTSEL	FB	R/W	0	0	0	0	0	0	0	0

Table 8-1. Register Values After RESET (continued)

Register Function	Register Name	Address	R/W	Bit Values After RESET							
				7	6	5	4	3	2	1	0
Port Status LSB	PXSTS0	FC	R	0	0	0	0	0	0	0	0
Port Status MSB	PXSTS1	FD	R	0	0	0	0	0	0	0	0
Port Status Change LSB	PXSTS2	FE	R	0	0	0	0	0	0	0	0
Port Status Change MSB	PXSTS3	FF	R	0	0	0	0	0	0	0	0
Lower 8 bits of the Frame Num. The Frame Number register contains the frame number received with the last SOF.	FRAMEL	B0	R	0	0	0	0	0	0	0	0
Upper bits of the Frame NUM	FRAMEH	B1	R	0	0	0	0	0	0	0	0
Interrupt Pending	INTPEND	B2	R/W	0	0	0	0	0	0	0	0
Interrupt Mask	INTMASK	B3	R/W	0	0	0	0	0	0	0	0
Function Endpoint 0 Data Buffer	FEP0BUF	B4	R/W	0	0	0	0	0	0	0	0
Function Endpoint 1 Data Buffer	FEP1BUF	B5	R/W	0	0	0	0	0	0	0	0
Function Endpoint 2 Data Buffer	FEP2BUF	B6	R/W	0	0	0	0	0	0	0	0
Function Interrupt Status	FINTSTS	B7	R/W	0	0	0	0	0	0	0	0
Function Interrupt Enable	FINTEN	B8	R/W	0	0	0	0	0	0	0	0
Function Endpoint Byte Count register, Valid for Transfers	FEPBCNT	B9	R	0	0	0	0	0	0	0	0
Endpoint In Control/Status	FEPINCSR	BA	R/W	0	0	0	0	0	0	0	0
Endpoint Out Control/Status	FEPOUTCSR	BB	R/W	0	0	0	0	0	0	0	0
Endpoint Configuration	FEPCFG	BC	R/W	0	0	0	0	0	0	0	0
Endpoint Max. Packet size	FEPMAXP	BD	R/W	0	0	0	1	0	0	0	0
Endpoint Select	FEPSEL	BE	R/W	0	0	0	0	0	0	0	0
Function USB Address	FADDR	BF	R/W	0	0	0	0	0	0	0	0
Port 0 control (LOW byte)	P0CONL	A0	R/W	0	0	0	0	0	0	0	0
Port 0 control (HIGH byte)	P0CONH	A1	R/W	0	0	0	0	0	0	0	0
Port 1 control (LOW byte)	P1CONL	A2	R/W	0	0	0	0	0	0	0	0
Port 1 control (HIGH byte)	P1CONH	A3	R/W	0	0	0	0	0	0	0	0
Port 2 control (LOW byte)	P2CONL	A4	R/W	0	0	0	0	0	0	0	0
Port 2 control (HIGH byte)	P2CONH	A5	R/W	0	0	0	0	0	0	0	0
Port 0 Interrupt Control	P0INT	A6	R/W	0	0	0	0	0	0	0	0
Port 0 Interrupt Pending	P0PND	A7	R/W	0	0	0	0	0	0	0	0
Port 2 Interrupt Control	P2INT	A8	R/W	0	0	0	0	0	0	0	0
Port 2 Interrupt Pending	P2PND	A9	R/W	0	0	0	0	0	0	0	0
Port 3 Control	P3CON	AA	R/W	0	0	0	0	0	0	0	0
Port 4 Control	P4CON	AB	R/W	0	0	0	0	0	0	0	0

Table 8-1. Register Values After RESET (concluded)

Register Function	Register Name	Address	R/W	Bit Values After RESET							
				7	6	5	4	3	2	1	0
Port 4 Interrupt Enable & Pending	P4INTn P4PNDn	AC	R/W	0	0	0	0	0	0	0	0
Port 2 Data	P2	AD	R/W	0	0	0	0	0	0	0	0
Port 3 Data	P3	AE	R/W	–	–	–	–	0	0	0	0
Port 4 Data	P4	AF	R/W	–	–	–	–	0	0	0	0
Location 90H - 95H are not mapped.											
USB Mode select	USBMODE	96	R/W	0	0	0	0	0	0	0	0
USB/GPIO Mode select	USBSEL	97	R/W	0	0	0	0	0	0	0	0
DM0/GPIO, DP0/GPIO Data (Only GPIO Mode)	GPIODATA	98	R/W	0	0	0	0	0	0	0	0
GPIO Control and Interrupt Pending	GPIOCONIN T	99	R/W	0	0	0	0	0	0	0	0
Timer B Counter (HIGH)	TBCNTH	9A	R	0	0	0	0	0	0	0	0
Timer B Counter (LOW)	TBCNTL	9B	R	0	0	0	0	0	0	0	0
Timer B Data (HIGH)	TBDATAH	9C	R/W	1	1	1	1	1	1	1	1
Timer B Data (LOW)	TBDATAL	9D	R/W	1	1	1	1	1	1	1	1
Timer B control	TBCON	9E	R/W	0	0	0	0	0	0	0	0
Timer Interrupt Priority Pending	INTPND	9F	R/W	–	–	–	–	0	0	0	0

NOTES

9

I/O PORTS

OVERVIEW

The KS86C6308/P6308 USB Mode has five I/O ports (0-4) with a total of 30 pins. GPIO mode has six I/O ports (0-4 and DP0/GPIO, DM0/GPIO) with a total of 32 pins.

You can access these ports directly by writing or reading port data register addresses.

For keyboard applications, ports 0, 1 and 2 are usually configured as keyboard matrix input/output. Port 3 can be configured as LED drive. Port 4 is used for host communication or for controlling a mouse or other external device.

Table 9-1. KS86C6308/P6308 Port Configuration Overview

Port	Function Description	Programmability
0	Bit-programmable I/O port for Schmitt trigger input or open-drain output. Port0 can be individually configured as external interrupt inputs. Pull-up resistors are assignable by software.	Bit
1	Bit-programmable I/O port for Schmitt trigger input or open-drain output. Pull-up resistors are assignable by software.	Bit
2	Bit-programmable I/O port for Schmitt trigger input or open-drain output. Port2 can be individually configured as external interrupt inputs. Pull-up resistors are assignable by software.	Bit
3	Bit-programmable I/O port for Schmitt trigger input, open-drain or push-pull output. P3.3 can be used to system clock output (CLO) pin and timer pin.	Bit
4	Bit-programmable I/O port for Schmitt trigger input or open-drain output or push-pull output. Port4 can be individually configured as external interrupt inputs. In output mode, pull-up resistors are assignable by software. But in input mode, pull-up resistors are fixed.	Bit
DP0/GPIO DM0/GPIO (GPIO mode Only)	Bit-programmable I/O port for Schmitt trigger input or open-drain output or push-pull output. This port individually configured as external interrupt inputs. In output mode, pull-up resistors are assignable by software. But in input mode, pull-up resistors are fixed.	Bit

PORT DATA REGISTERS

Table 9-2 gives you an overview of the port data register names, locations and addressing characteristics. Data registers for ports 0-4 have the structure shown in Figure 9-1.

Table 9-2. Port Data Register Summary

Register Name	Mnemonic	Decimal	Hex	R/W
Port 0 data register	P0	224	E0H	R/W
Port 1 data register	P1	225	E1H	R/W
Port 2 data register	P2	173	AD	R/W
Port 3 data register	P3	174	AE	R/W
Port 4 data register	P4	175	AF	R/W

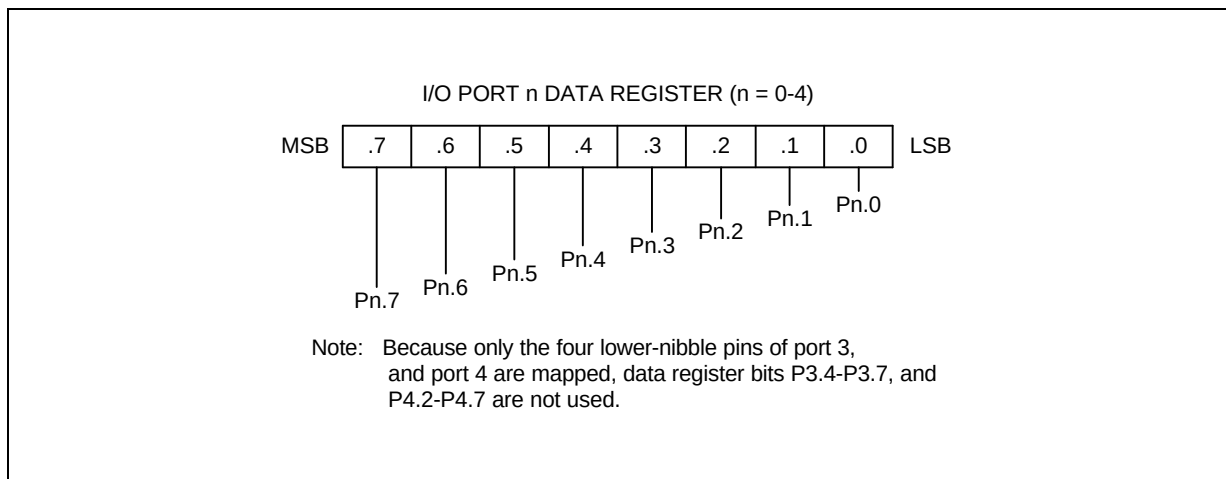


Figure 9-1. Port Data Register Format

PORT 0 AND PORT 1

Ports 0 bit-programmable, general-purpose, I/O ports. You can select Schmitt trigger input mode, N-CH open drain output mode.

You can access ports 0 and 1 directly by writing or reading the corresponding port data registers — P0 (E0H) and P1 (E1H). A reset clears the port control registers P0CONH, P0CONL, P1CONH and P1CONL to '00H', configuring all port 0 and port 1 pins as Schmitt trigger inputs.

In typical keyboard controller applications, the sixteen port 0 and port 1 pins can be used to check pressed key from keyboard matrix by generating keystroke output signals.

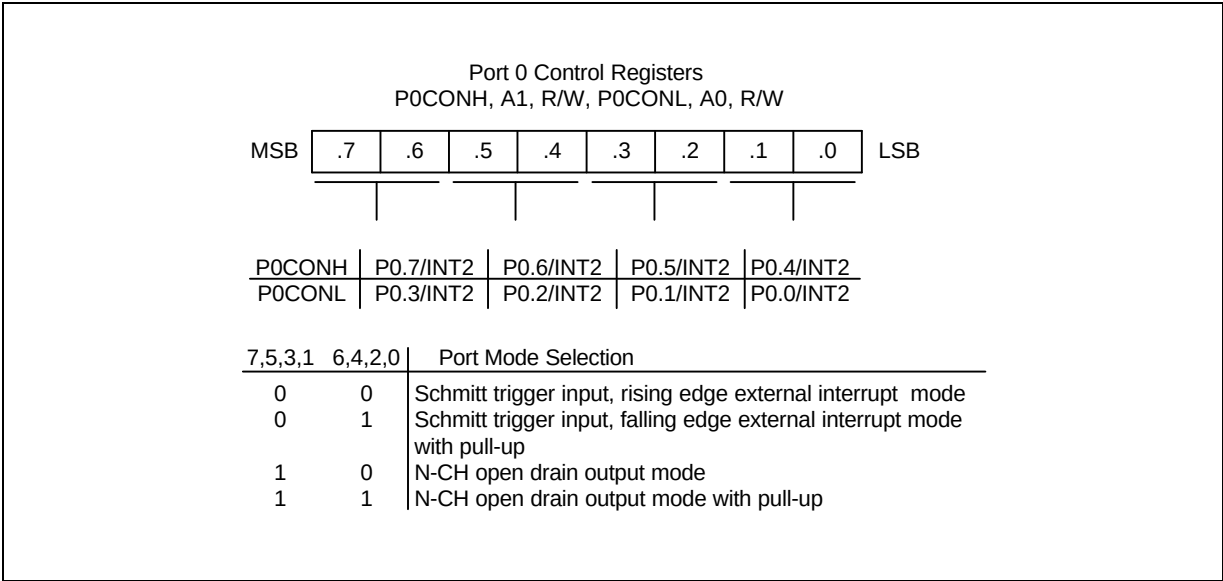


Figure 9-2. Port 0 Control Registers (P0CONH, P0CONL)

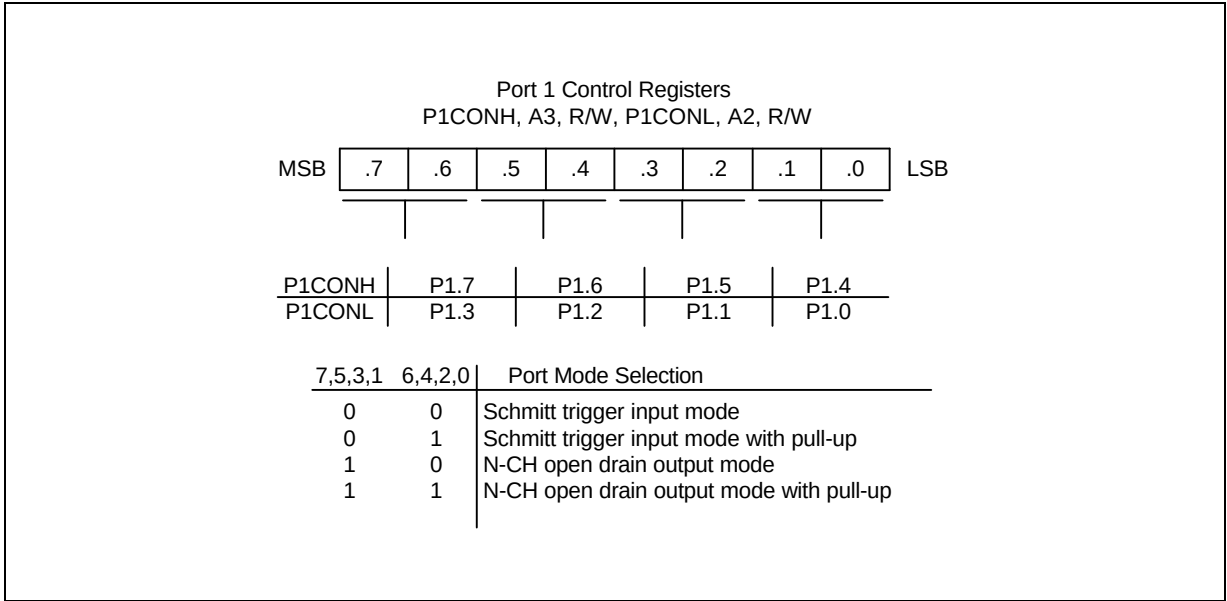


Figure 9-3. Port 1 Control Registers (P1CONH, P1CONL)

PORT 2

Port 2 is an 8-bit I/O port with individually configurable pins. It can be used for general I/O (Schmitt trigger input mode or push-pull output mode). Or, you can use port 2 pins as external interrupt (INT0) inputs. In addition, you can configure a pull-up resistor to individual pins using control register settings. All port 2 pin circuits have noise filters.

In typical keyboard controller applications, the port 2 pins are programmed to receive key input data from the keyboard matrix.

You can address port 2 bits directly by writing or reading the port 2 data register, P2 (AD). The port 2 high-byte and low-byte control registers, P2CONH and P2CONL, are located at addresses EA and A4, respectively.

Two additional registers, are used for interrupt control: P2INT (A8) and P2PND (A7). By setting bits in the port 2 interrupt enable register P2INT, you can configure specific port 2 pins to generate interrupt requests when rising or falling signal edges are detected. The application program polls the port 2 interrupt pending register, P2PND, to detect interrupt requests. When an interrupt request is acknowledged, the corresponding pending bit must be cleared by the interrupt service routine.

In case of keyboard applications, the port 2 pins can be used to read key value from key matrix.

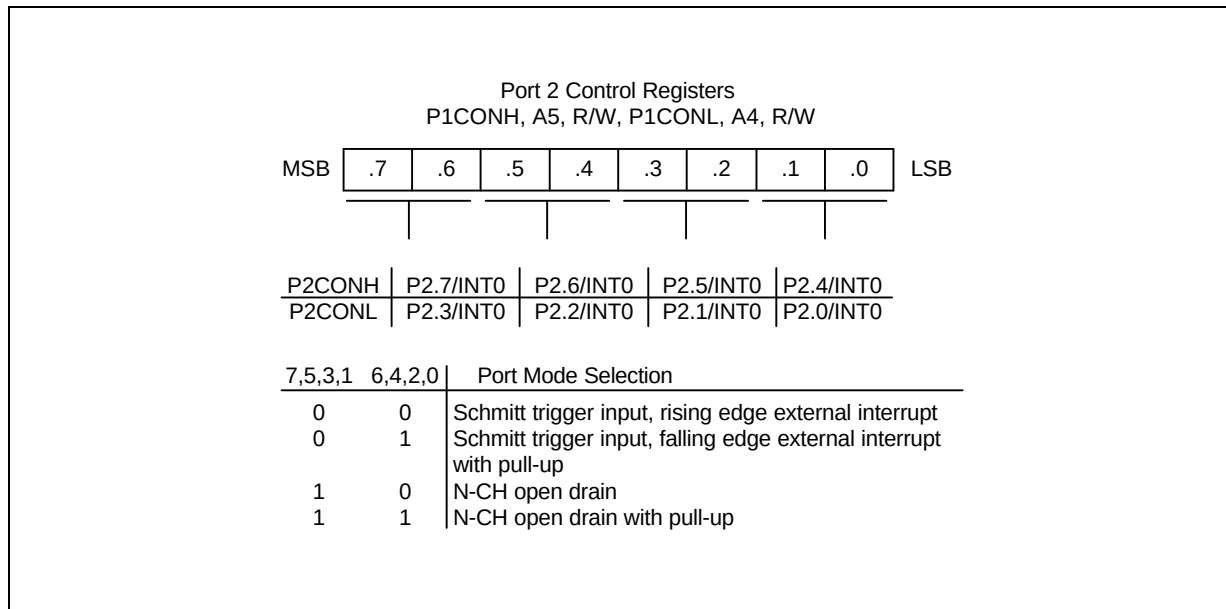


Figure 9-4. Port 2 Control Registers (P2CONH, P2CONL)

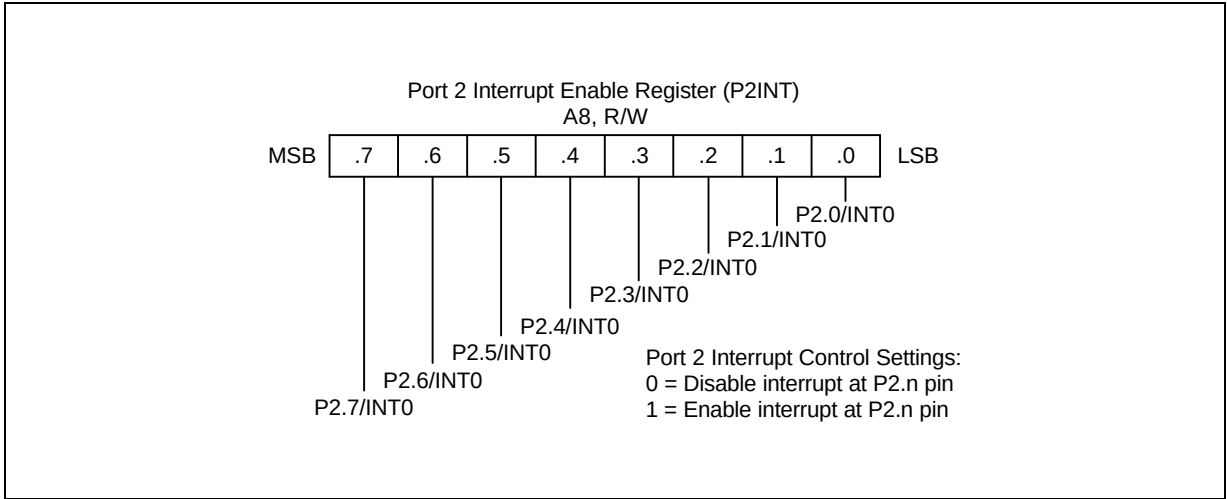


Figure 9-5. Port 2 Interrupt Enable Register (P2INT)

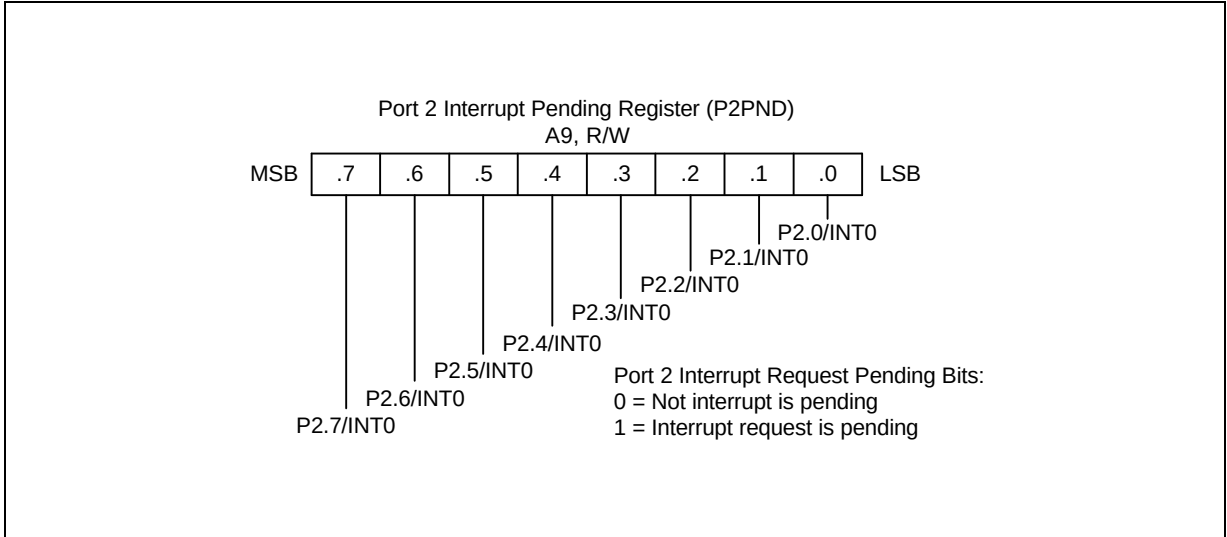


Figure 9-6. Port 2 Interrupt Pending Register (P2PND)

PORT 3

Port 3 is a 4-bit, bit-configurable, general I/O port. It is designed for high-current functions such as LED drive.

A reset configures P3.0-P3.3 to Schmitt trigger input mode. Using the P3CON register (A4), you can alternatively configure the port 3 pins as n-channel, open-drain outputs. P3.3 can be used to system clock output (CLO) port and Timer A, Timer B port.

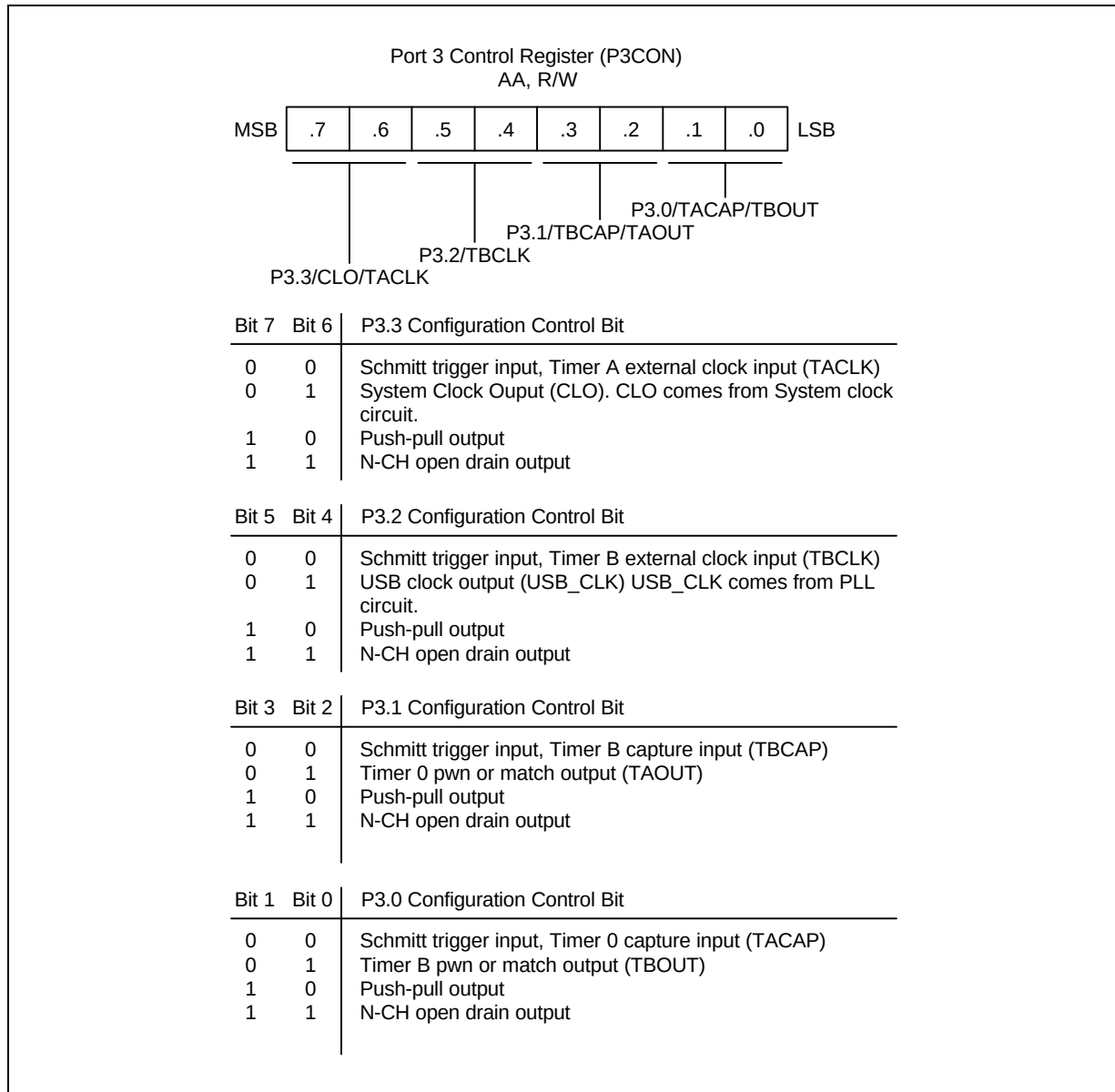


Figure 9-7. Port 3 Control Register (P3CON)

PORT 4

Port 4 is a 4-bit I/O port with individually configurable pins. It can be used for general I/O (Schmitt trigger, N-CH open drain output mode, push-pull output mode). Or, you can use port 4 pins as external interrupt (INT1) inputs. In addition, you can configure a pull-up resistor to individual pins using control register settings. All port 4 pins have noise filters.

A reset configures P4.0-P4.1 to input mode. You address port 4 directly by writing or reading the port 4 data register, P4 (AF). The port 4 control register, P4CON, is located at AB.

A additional registers used for interrupt control: P4INTPND (AC). By setting bits in the port 4 interrupt enable and pending register P4INTPND.7-P4INTPND.4, you can configure specific port 4 pins to generate interrupt requests when falling signal edges are detected. The application program polls the interrupt pending register, P4INTPND.1-P4INTPND.0, to detect interrupt requests. When an interrupt request is acknowledged, the corresponding pending bit must be cleared by the interrupt service routine.

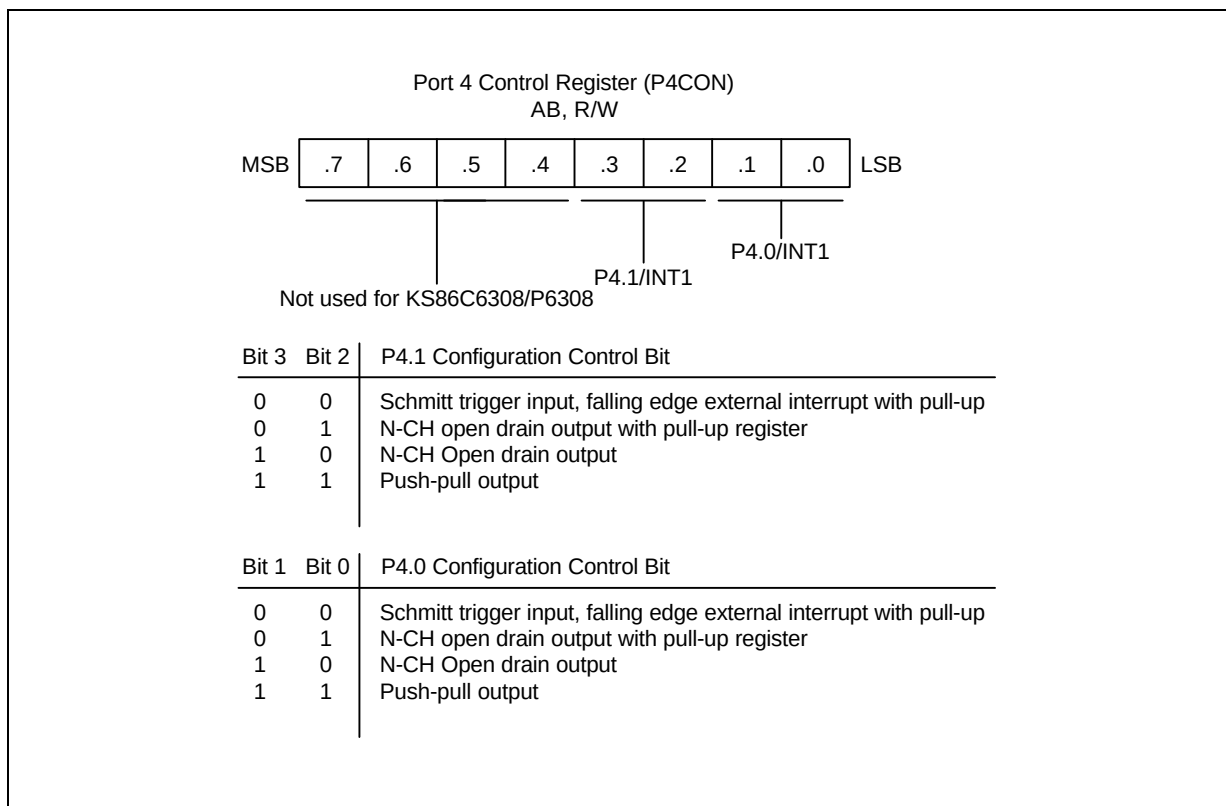


Figure 9-8. Port 4 Control Register (P4CON)

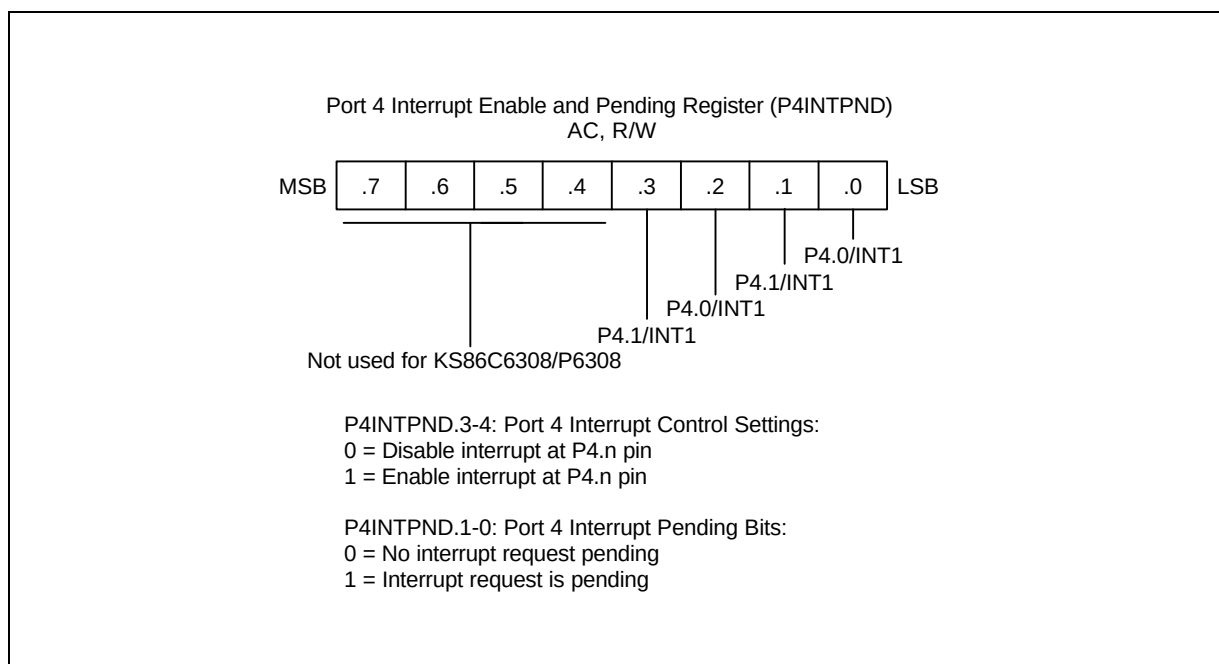


Figure 9-9. Port 4 Interrupt Enable and Pending Register (P4INTPND)

DP0/GPIO, DM0/GPIO

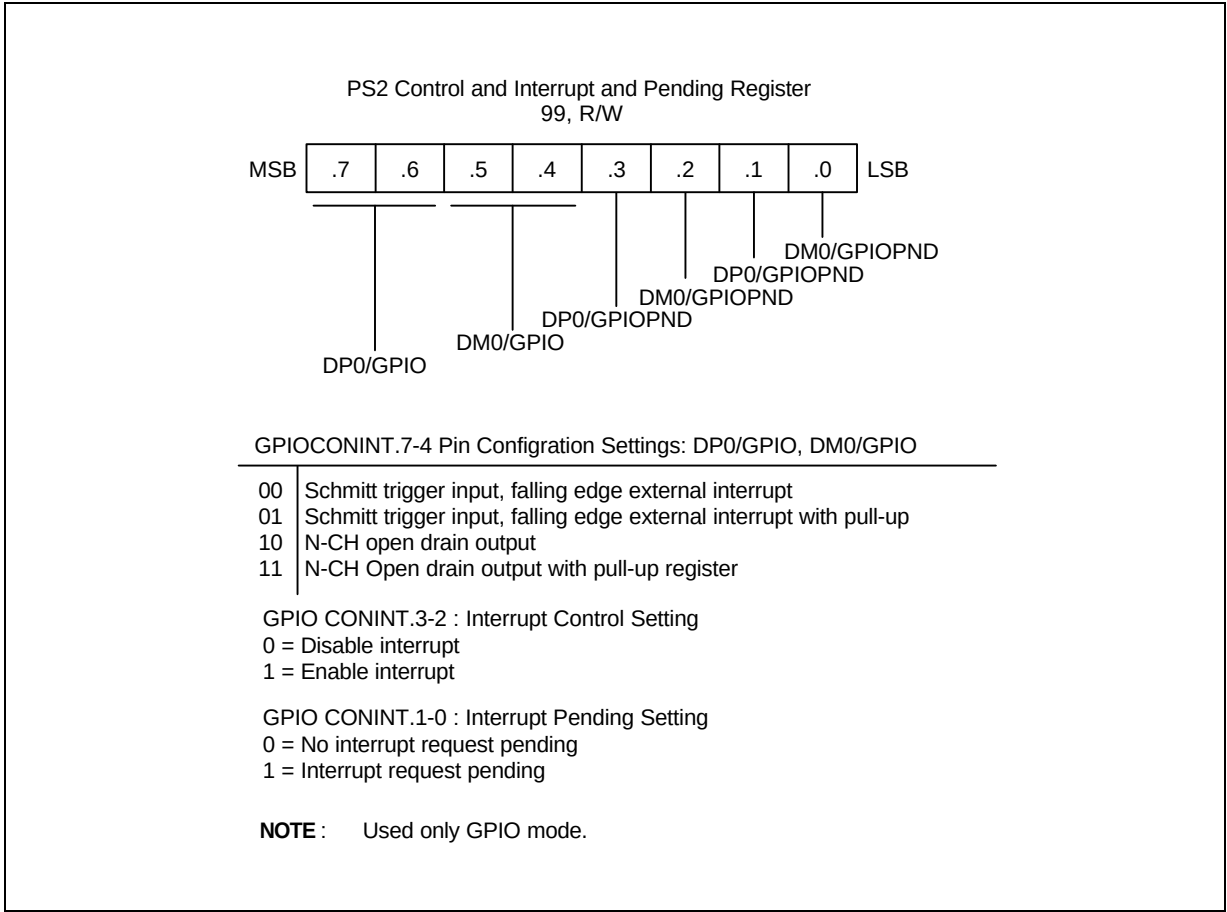


Figure 9-10. GPIOCONINT Register (GPIOCONINT)

10

BASIC TIMER and TIMER A, TIMER B

MODULE OVERVIEW

The KS86C6308/P6308 has two default timers: an 8-bit *basic timer* and one 8-bit general-purpose timer/counter. The 8-bit timer/counter is called *timer A*.

Basic Timer (BT)

You can use the basic timer (BT) in two different ways:

- As a watchdog timer to provide an automatic reset mechanism in the event of a system malfunction.
- To signal the end of the required oscillation stabilization interval after a reset or a Stop mode release.

The functional components of the basic timer block are:

- Clock frequency divider (f_{OSC} divided by 4096, 1024, or 128) with multiplexer
- 8-bit basic timer counter, BTCNT (DDH, read-only)
- Basic timer control register, BTCON (DCH, read/write)

BASIC TIMER CONTROL REGISTER (BTCON)

The basic timer control register, BTCON, is used to select the input clock frequency, to clear the basic timer counter and frequency dividers, and to enable or disable the watchdog timer function.

A reset clears BTCON to '00H'. This enables the watchdog function and selects a basic timer clock frequency of $f_{OSC}/4096$. To disable the watchdog function, you must write the signature code '1010B' to the basic timer register control bits BTCON.7-BTCON.4.

The 8-bit basic timer counter, BTCNT, can be cleared at any time during normal operation by writing a "1" to BTCON.1. To clear the frequency dividers for both the basic timer input clock and the timer 0 clock, you write a "1" to BTCON.0.

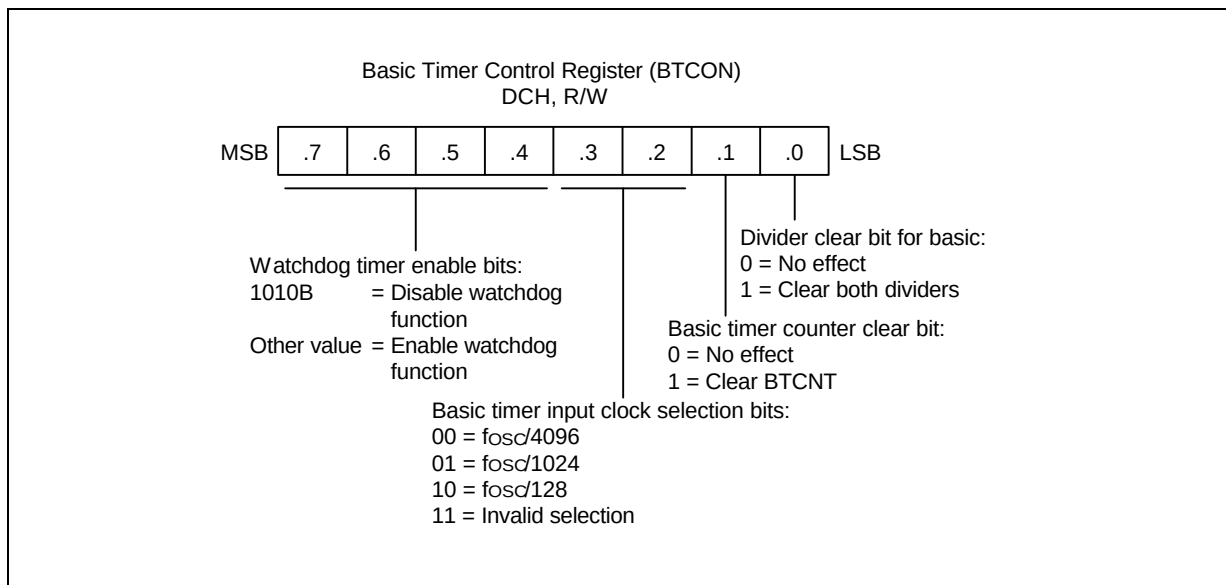


Figure 10-1. Basic Timer Control Register (BTCON)

BASIC TIMER FUNCTION DESCRIPTION

Watchdog Timer Function

You can program the basic timer overflow signal to generate a reset by setting BTCON.7-BTCON.4 to any value other than '1010B' (The '1010B' value disables the watchdog function). A reset clears BTCON to '00H', automatically enabling the watchdog timer function. A reset also selects the CPU clock (as determined by the current CLKCON register setting) divided by 4096 as the BT clock.

A reset whenever a basic timer counter overflow occurs. During normal operation, the application program must prevent the overflow, and the accompanying reset operation, from occurring. To do this, the BTCNT value must be cleared (by writing a "1" to BTCON.1) at regular intervals.

If a system malfunction occurs due to circuit noise or some other error condition, the BT counter clear operation will not be executed and a basic timer overflow will occur, initiating a reset. In other words, during normal operation, the basic timer overflow loop (a bit 7 overflow of the 8-bit basic timer counter, BTCNT) is always broken by a BTCNT clear instruction. If a malfunction does occur, a reset is triggered automatically.

Oscillation Stabilization Interval Timer Function

You can also use the basic timer to program a specific oscillation stabilization interval following a reset or when Stop mode has been released by an external interrupt.

In Stop mode, whenever a reset or an external interrupt occurs, the oscillator starts. The BTCNT value then starts increasing at the rate of $f_{OSC}/4096$ (for reset), or at the rate of the preset clock source (for an external interrupt). When BTCNT.4 is set, a signal is generated to indicate that the stabilization interval has elapsed and to gate the clock signal off to the CPU so that it can resume normal operation.

In summary, the following events occur when Stop mode is released:

1. During Stop mode, a power-on reset or an external interrupt occurs to trigger the Stop mode release and oscillation starts.
2. If a power-on reset occurred, the basic timer counter will increase at the rate of $f_{OSC}/4096$. If an external interrupt is used to release Stop mode, the BTCNT value increases at the rate of the preset clock source.
3. Clock oscillation stabilization interval begins and continues until bit 4 of the basic timer counter is set.
4. When a BTCNT.4 is set, normal CPU operation resumes.

Figures 10-2 and 10-3 shows the oscillation stabilization time on RESET and STOP mode release

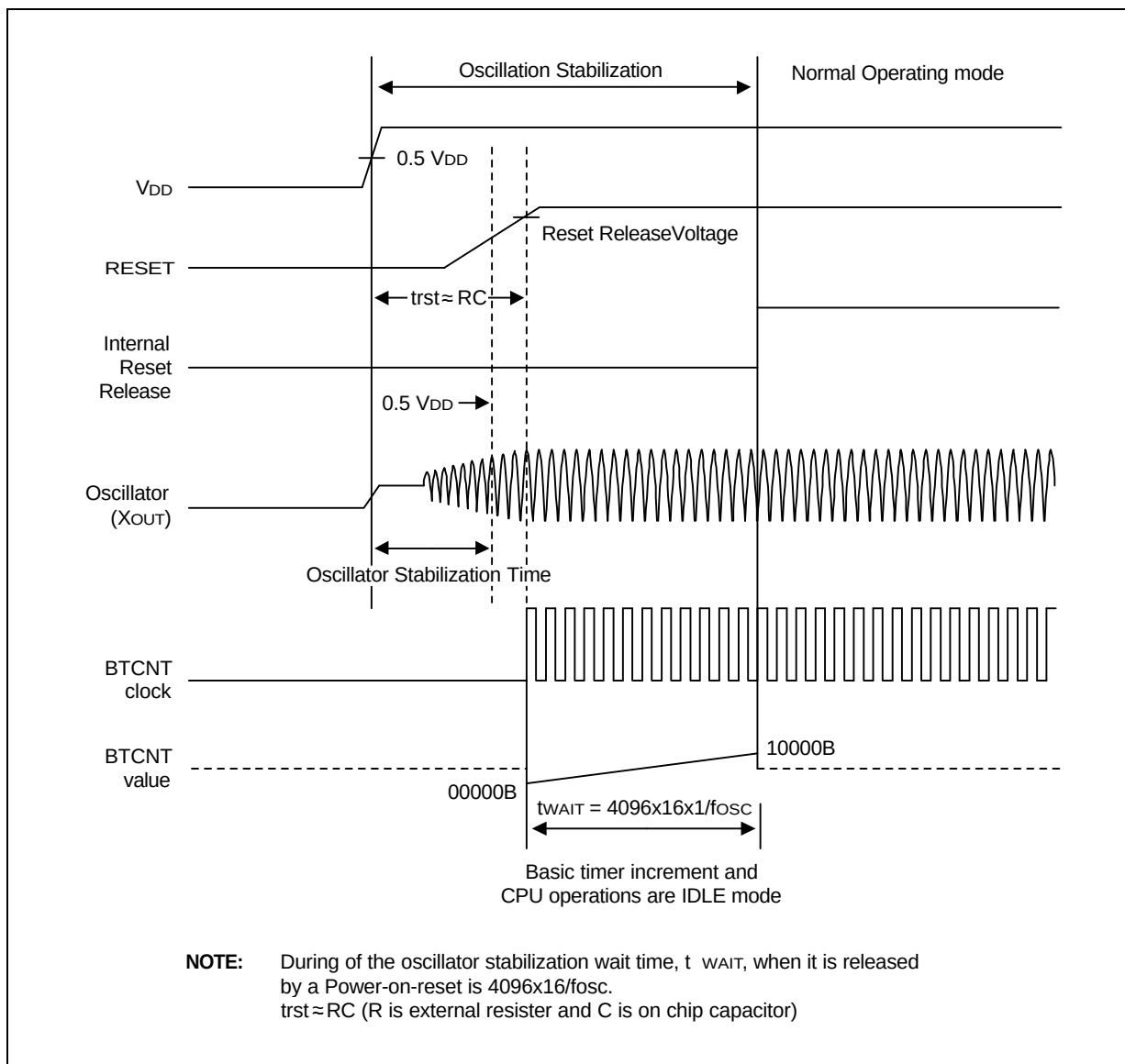
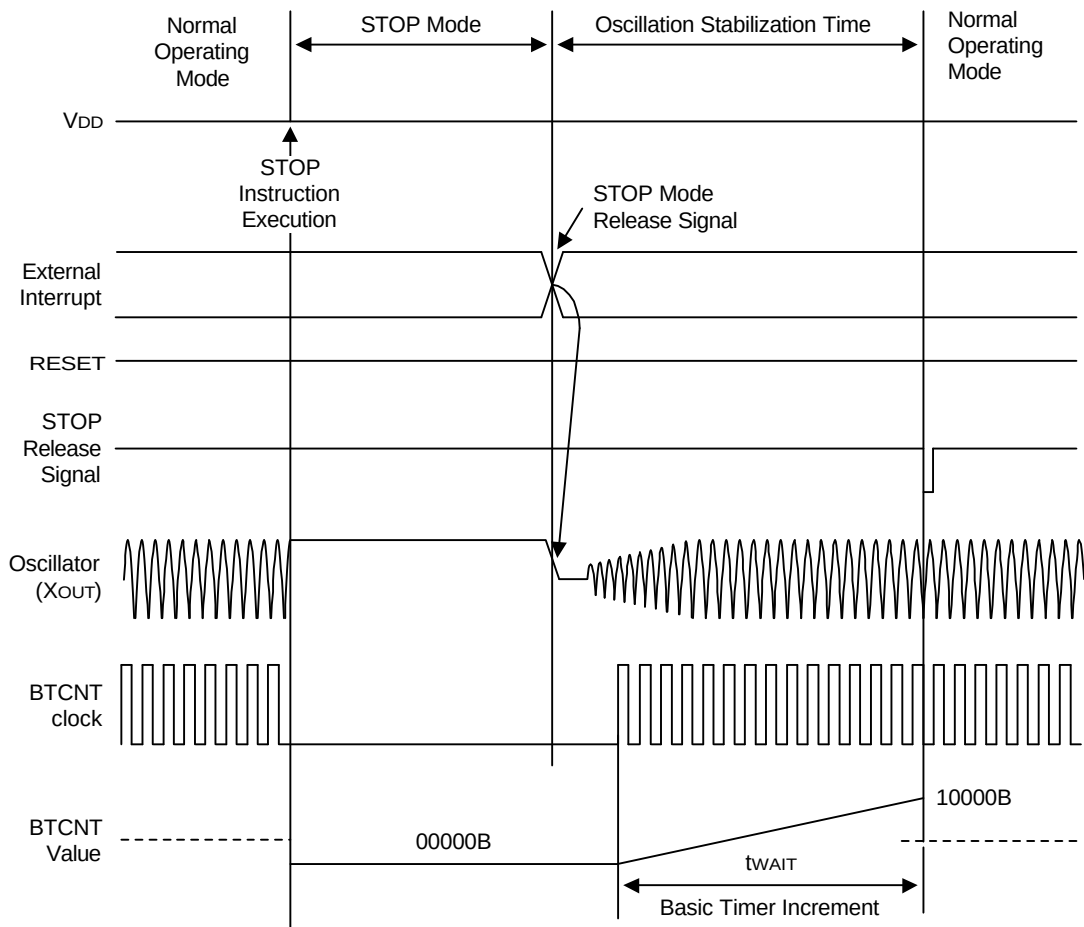


Figure 10-2. Oscillation Stabilization Time on RESET



NOTE: Duration of the oscillator stabilization wait time, t_{WAIT} , it is released by an interrupt is determined by the setting in basic timer control register, BTCON.

BTCON.3	BTCON.2	t_{WAIT}	t_{WAIT} (When f_{osc} is 10 MHz)
0	0	$(4096 \times 16)/f_{osc}$	10.92 ms
0	1	$(1024 \times 16)/f_{osc}$	2.7 ms
1	0	$(128 \times 16)/f_{osc}$	0.34 ms
1	1	Invalid setting	—

Figure 10-3. Oscillation Stabilization Time on STOP Mode Release

8-BIT TIMER A

OVERVIEW

The 8-bit timer A is an 8-bit general-purpose timer/counter. Timer A has three operating modes, one of which you select using the appropriate TACON setting:

- Capture input mode with a rising or falling edge trigger at the TACAP pin
- Interval timer mode (Toggle output at TAOUT pin)
- PWM mode (TAPWM)

Timer A has the following functional components:

- Clock frequency divider (fx divided by 1024, 256, or 64) with multiplexer
- External clock input pin (TACLK)
- 8-bit counter (TACNT), 8-bit comparator, and 8-bit reference data register (TADATA)
- I/O pins for capture input (TACAP) or PWM or match output (TAPWM, TAOUT)
- Timer A overflow interrupt (IRQ, vector, 00H) and match/capture interrupt generation (IRQ, vector, 00H)
- Timer A control register, TACON (D2H, r/w)

FUNCTION DESCRIPTION

The timer A module can generate two interrupts: the timer A overflow interrupt, and the timer A match/ capture interrupt (TAINT). A timer A overflow interrupt pending condition is automatically cleared by hardware when it has been serviced. A timer A match/capture interrupt, TAINT pending condition is also cleared by hardware when it has been serviced.

Interval Timer Function

The timer A module can generate an interrupt: the timer A match interrupt (TAINT). When timer A measure interrupt occurs and is serviced by the CPU, the pending condition is cleared automatically by hardware. In interval timer mode, a match signal is generated and TAOUT is toggled when the counter value is identical to the value written to the reference data register, TADATA. The match signal generates a timer A match interrupt and clears the counter.

If, for example, you write the value 10H to TADATA and 0AH to TACON, the counter will increment until it reaches 10H. At this point, the TA interrupt request is generated, the counter value is reset, and counting resumes.

Pulse Width Modulation Mode

Pulse width modulation (PWM) mode lets you program the width (duration) of the pulse that is output at the TAPWM pin. As in interval timer mode, a match signal is generated when the counter value is identical to the value written to the timer A data register.

In PWM mode, however, the match signal does not clear the counter. Instead, it runs continuously, overflowing at FFH, and then continues incrementing from 00H.

Although you can use the match signal to generate a timer A overflow interrupt, interrupts are not typically used in PWM-type applications. Instead, the pulse at the TAPWM pin is held to Low level as long as the reference data value is less than or equal to ($\leq$) the counter value and then the pulse is held to High level for as long as the data value is greater than ($>$) the counter value. One pulse width is equal to $t_{CLK} \times 256$.

Capture Mode

In capture mode, a signal edge that is detected at the TACAP pin opens a gate and loads the current counter value into the TA data register. You can select rising or falling edges to trigger this operation.

Timer A also gives you capture input source: the signal edge at the TACAP pin. You select the capture input by setting the value of the timer A capture input selection bit in the port 3 control register, P3CON.

When P3CON.1.0 is 00, the TACAP input or normal input is selected. When P3CON.3.2 is set to 01, TAOUT output is selected.

Both kinds of timer A interrupts can be used in capture mode: the timer A overflow interrupt is generated whenever a counter overflow occurs; the timer A match/capture interrupt is generated whenever the counter value is loaded into the TA data register. By reading the captured data value in TADATA, and assuming a specific value for the timer A clock frequency, you can calculate the pulse width (duration) of the signal that is being input at the TACAP pin.

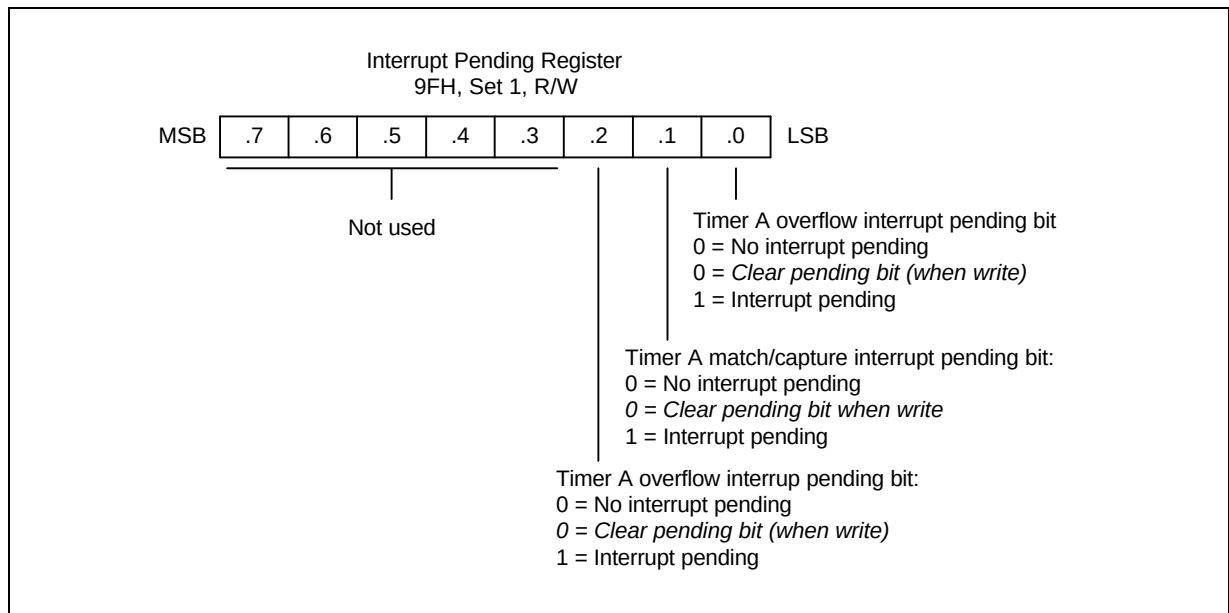


Figure 10-5. Interrupt Pending Register

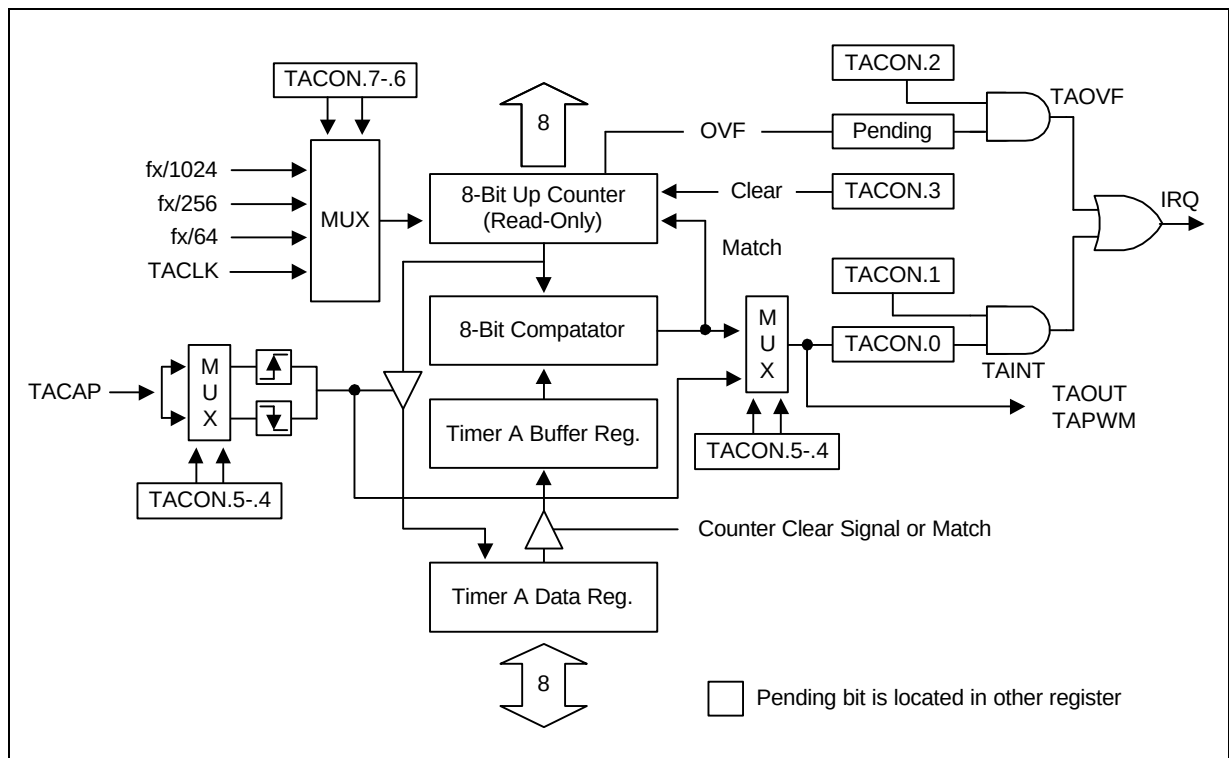


Figure 10-6. Interrupt Pending Register

16-BIT TIMER B

OVERVIEW

The 16-bit timer B is a 16-bit general-purpose timer/counter. Timer B has three operating modes, one of which you select using the appropriate TBCON setting:

- Interval timer mode(Toggle output at TBOUT pin)
- Capture input mode with a rising or falling edge trigger at the TBCAP pin
- PWM mode (TBPWM)

Timer B has the following functional components:

- Clock frequency divider (fxx divided by 1024,256, 64, 8 or 1) with multiplexer
- External clock input pin (TBCLK)
- 16-bit counter (TBCNTH/L), 16-bit comparator, and 16-bit reference data register (TBDATAH/L)
- I/O pins for capture input (TBCAP), or PWM or match output(TBPWM, TBOUT)
- Timer B overflow interrupt and match/capture interrupt generation
- Timer B control register, TBCON (set 1, 9EH, read/write)

FUNCTION DESCRIPTION

Timer B Interrupts. The timer B module can generate two interrupts: the timer B overflow interrupt (TBOVF), and the timer B match/ capture interrupt (TBINT). TBOVF is interrupt level IRQ, vector 00H. TBINT also belongs to interrupt level IRQ.

A timer B overflow interrupt pending condition is automatically cleared by hardware when it has been serviced. A timer B match/capture interrupt, TBINT pending condition is also cleared by hardware when it has been serviced.

Interval Timer Function

The timer B module can generate an interrupt: the timer B match interrupt (TBINT).

TBINT is assigned the vector address, 00H. When a timer B measure interrupt occurs and is serviced by the CPU, the pending condition is cleared automatically by hardware.

In interval timer mode, a match signal is generated and TBOUT is toggled when the counter value is identical to the value written to the timer B reference data register, TBDATAH/L. The match signal generates a timer B match interrupt and clears the counter.

If, for example, you write the value 00B0H to TBDATAH/L and 06H to TBCON, the counter will increment until it reaches 00B0H. At this point, the TB interrupt request is generated, the counter value is reset, and counting resumes.

Pulse Width Modulation Mode

Pulse width modulation (PWM) mode lets you program the width (duration) of the pulse that is output at the TBPWM pin. As in interval timer mode, a match signal is generated when the counter value is identical to the value written to the timer B data register. In PWM mode, however, the match signal does not clear the counter but can generate a match interrupt. The counter runs continuously, overflowing at FFFFH, and then repeat the incrementing from 0000H.

Whenever an overflow is occurred, an overflow(OVF) interrupt can be generated.

Although you can use the match or the overflow interrupt in PWM mode, interrupts are not typically used in PWM-type applications. Instead, the pulse at the TBPWM pin is held to Low level as long as the reference data value is less than or equal to ($\leq$) the counter value and then the pulse is held to High level for as long as the data value is greater than ($>$) the counter value. One pulse width is equal to $t_{CLK} \times 65536$.

Capture Mode

In capture mode, a signal edge that is detected at the TBCAP pin opens a gate and loads the current counter value into the TB data register. You can select rising or falling edges to trigger this operation. Timer B also gives you capture input source: the signal edge at the TBCAP pin.

You select the capture input by setting the value of the timer B capture input selection bit in the port B control register, P3CON, (set 1, AAH). When P3CON3.2 is 00, the TBCAP input or normal input is selected. When P3CON3.2 is set to 11, normal output is selected.

Both kinds of timer 1 interrupts can be used in capture mode: the timer B overflow interrupt is generated whenever a counter overflow occurs; the timer B match/capture interrupt is generated whenever the counter value is loaded into the TB data register.

By reading the captured data value in TBDATAH/L, and assuming a specific value for the timer B clock frequency, you can calculate the pulse width (duration) of the signal that is being input at the TBCAP pin.

TIMER B CONTROL REGISTER (TBCON)

You use the timer B control register, TBCON, to

- Select the timer B operating mode (interval timer, capture mode, or PWM mode)
- Select the timer B input clock frequency
- Clear the timer B counter, TBCNTH/L
- Enable the timer B overflow interrupt or timer B match/capture interrupt
- Clear timer B match/capture interrupt pending conditions

TBCON is located in set 1 and at address 9EH, and is read/write addressable using Register addressing mode. A reset clears TBCON to '00H'. This sets timer B to normal interval timer mode, selects an input clock frequency of $f_{OSC}/1024$, and disables all timer B interrupts. To disable the counter operation, please set TBCON.7 -.5 to 111B.

You can clear the timer B counter at any time during normal operation by writing a "1" to TBCON.3. The timer B overflow interrupt (TBOVF) is interrupt level and has the vector address 00H. When a timer B overflow interrupt occurs and is serviced by the CPU, the pending condition is cleared automatically by hardware.

To enable the timer B match/capture interrupt, you must write TBCON.1 to "1". To generate the exact time interval, you should write TBCON.2 which clear counter and interrupt pending bit.

To detect a match/capture or overflow interrupt pending condition when TBINT or TBOVF is disabled, the application program should poll the pending bit. When a "1" is detected, a timer B match/capture or overflow interrupt is pending.

When the sub-routine has been serviced, the pending condition must be cleared by software by writing a "0" to the interrupt pending bit.

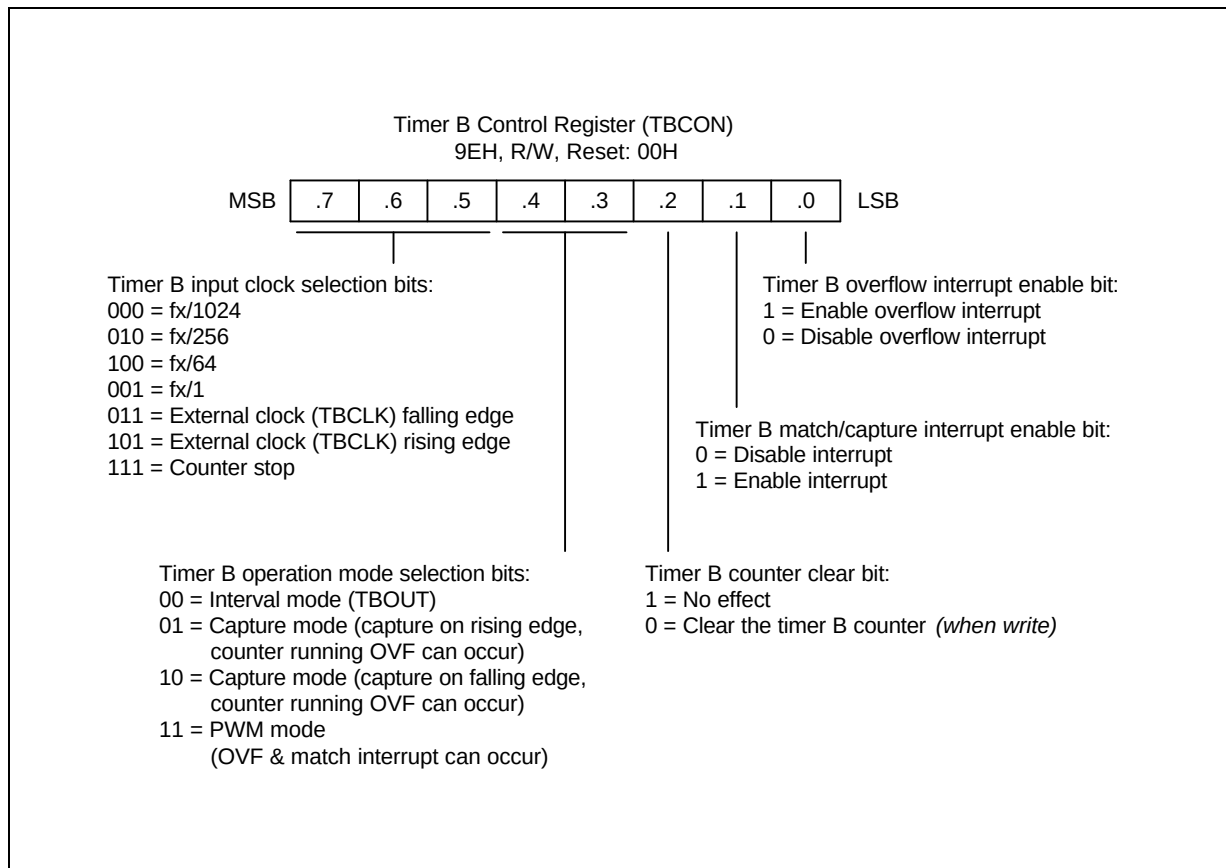


Figure 10-7. Timer B Control Register (TBCON)

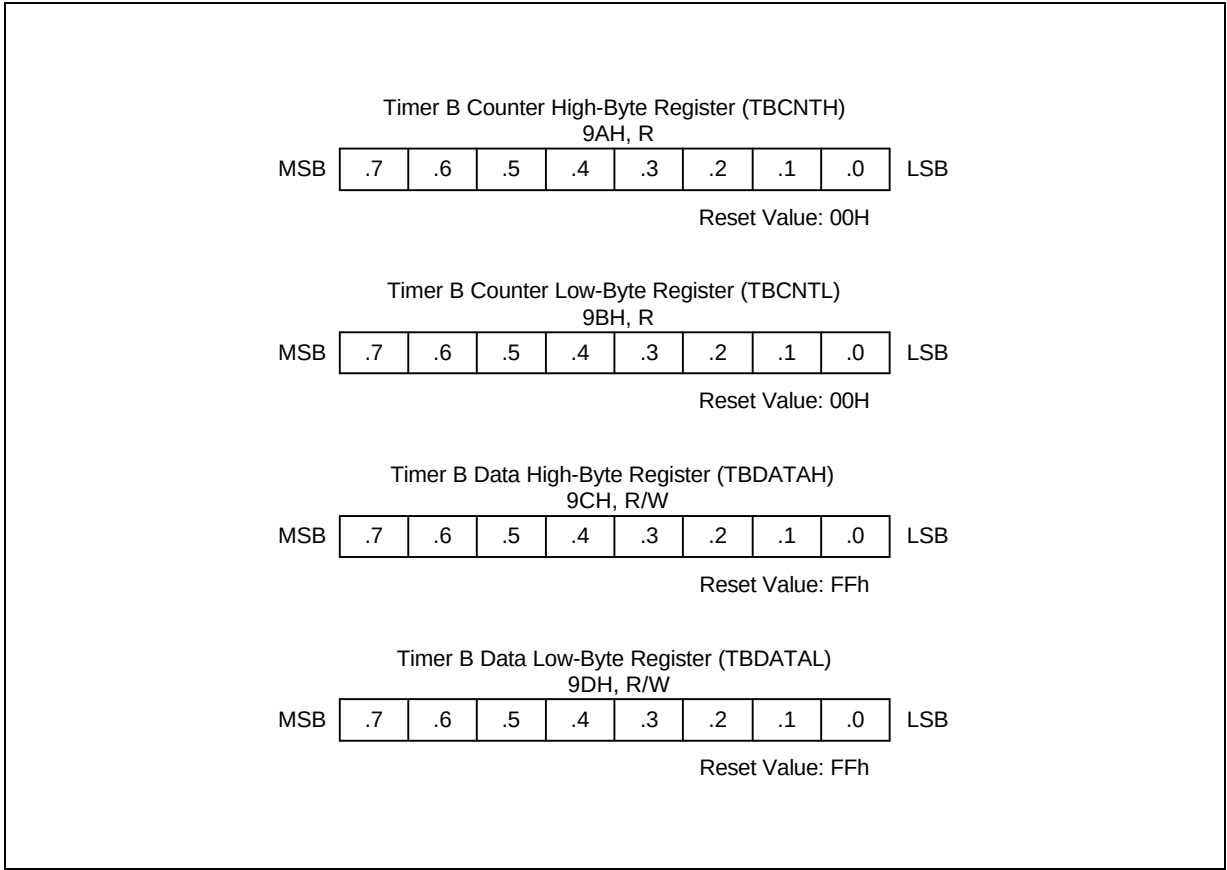


Figure 10-8. Timer B Counter Register

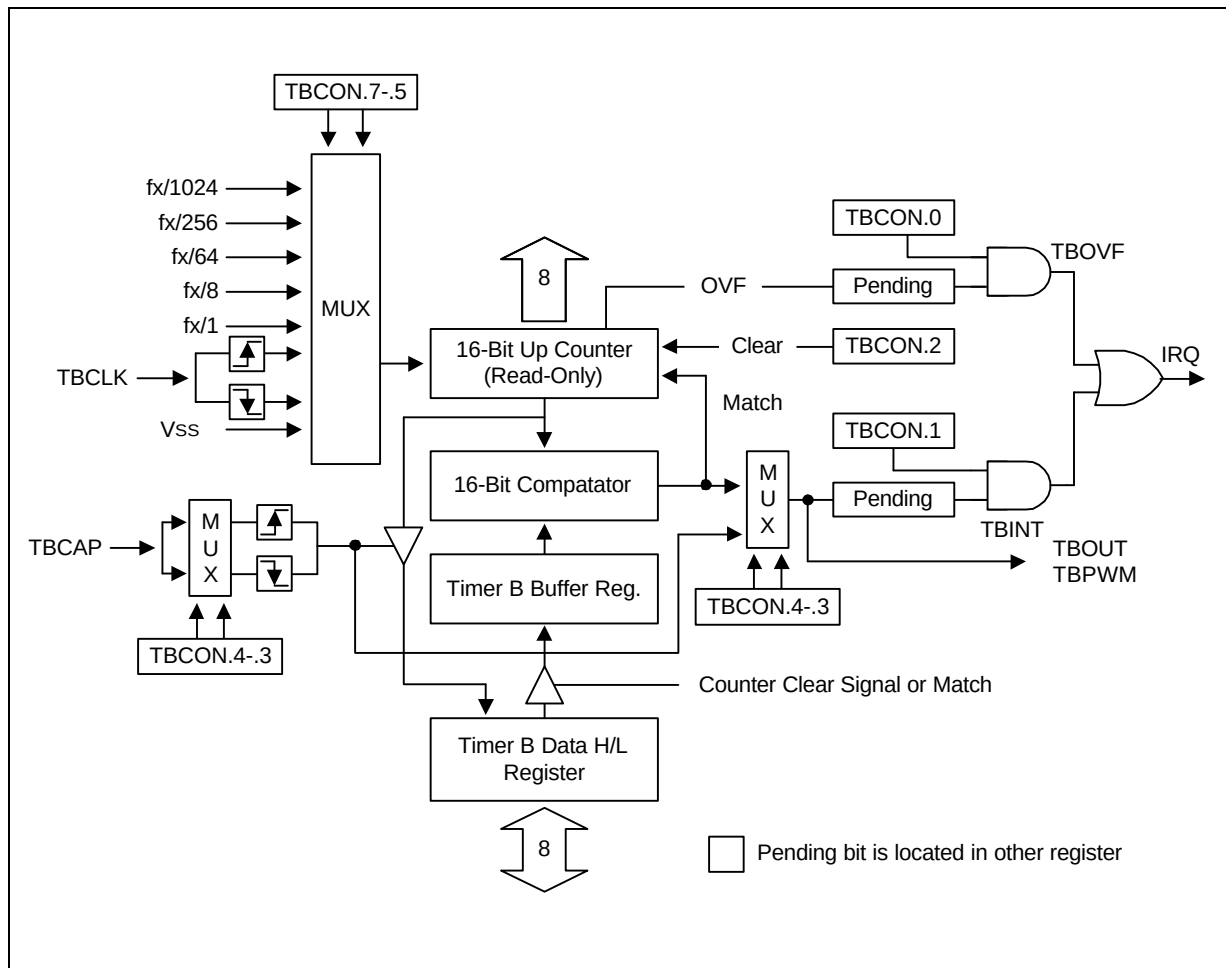


Figure 10-9. Timer B Functional Block Diagram

NOTES

11

UNIVERSAL SERIAL BUS

OVERVIEW

Universal Serial Bus (USB) is a communication architecture that supports data transfer between a host computer and a wide range of PC peripherals. USB is actually a cable bus in which the peripherals share its bandwidth through a host scheduled token based protocol. The basic blocks used in KS86C6308 are the HUB Repeater, HUB Controller, Serial Interface Engine and the Function Interface Unit.

The USB module in KS86C6308 can accommodate both full speed (12Mbps) and low speed (1.5Mbps) transfer rate USB device as described in the Universal Serial Bus Specification Revision 1.0. KS86C6308 can be briefly describe as a microcontroller with SAM 88RCRI core with an on-chip USB peripheral that can serve as a USB function and a HUB as can be seen in Figure 11-1.

The KS86C6308 is a compound USB device. Its usability is dynamic, it can serve as a function by providing an interface for a PC peripheral and at the same time, it can serve as a hub by providing USB ports for additional PC peripherals or it can perform these functions separately.

Please refer to the USB specification revision 1.0 for detail description of USB.

USB Functional Features:

- Three functions endpoint pairs
- FIFO sizes; endpoint 1 and 2 support three transfer types (8, 16, 64 bytes)
- Connectivity to four downstream ports
- Power management
- Device connect/disconnect detection
- Bus fault detection and recovery
- Supports both full speed(12Mbps) and low speed (1.5Mbps) transfer rate
- Bus powered support
- Programmable ganged and individual port power control
- Protocol handling in hardware
- Hub control command execution in hardware
- Flexible hub unit interface to MCU for descriptors and other extensions
- Embedded function with two programmable endpoints
- Built-in 3.3V voltage regulator

General Function Features:

- On-chip USB transceivers
- Automatic transmit/receive FIFO management
- Suspend/resume
- USB interrupt
- Programmable ganged power switching and individual port power switching

General HUB Features:

Most HUB functions will be carried out by hardwired USB module except for few special instructions.

- HUB endpoint0: transmit/receive FIFO (8-bytes)
- HUB endpoint1: one transmit data buffer register (1-byte)
- One internal downstream port
- Four external downstream port

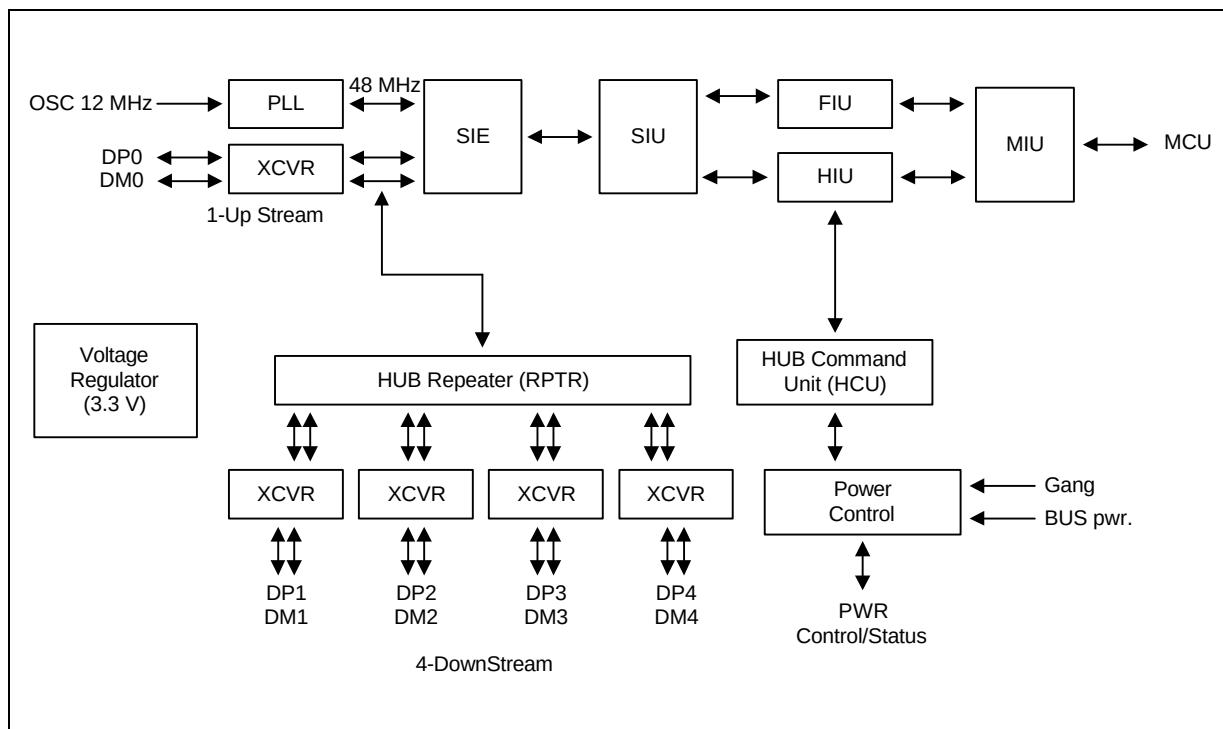


Figure 11-1. USB Module Block Diagram

FUNCTION DESCRIPTIONS

SERIAL BUS INTERFACE ENGINE (SIE)

The Serial Interface Engine handles the communication protocol of the USB, it is shared between the HUB Controller and the Function Interface as only one can transmit or receive at a time. The function of the SIE is to carry out all of the USB serial protocol, data encoding/decoding, CRC checking/generation and data conversion between serial and parallel data.

The Serial Interface Engine implements the protocol layer of the USB. It handles the clock recovery, error checking, the handshake on the USB bus if the packet was directed to it, bus time out if response from the host is late, and all other USB protocol related functions. It consists of Digital Phase Locked Loop (DPLL) for clock recovery from the incoming data, CRC checker and generator, bit stuff and bit removal logic, NRZI encoder/decoder, shift register for serial/parallel conversion, PID decoder, data toggle and sync detect logic.

SIE INTERFACE UNIT (SIU)

The SIE Interface Unit interfaces with SIE to get the parallel data and pass on to the FIU and HIU, and to multiplex the data from the FIU and HIU and send it to SIE. Other important function of the SIU is to compare the device and endpoint address in the token packet with the valid device and endpoint addresses from the hub and the embedded function, and generate an address valid signal to the SIE so it can complete the handshake and start to wait for the data phase.

HUB REPEATER (RPTR)

The HUB Repeater function is to repeat the data from the upstream port to all enabled downstream ports and to repeat the data from one of the downstream port to the upstream port. Multiple downstream ports cannot be transmitting data at the same time according to the USB protocol, if that happens it would be an error and will be handled separately. This block handles the global suspend and resume from the upstream port and also handles the remote resume from a downstream port to the upstream port, as well as to all the enable downstream port.

HUB Repeater block consists of logic for port control, repeater state machine and data multiplexers for data out to upstream and each downstream ports. The port control manages the connectivity/disconnectivity, suspend/resume, reset and SOP/EOP detection. The repeater state-machine manages the direction of the data multiplexers, and detection of error condition like babble or loss of activity.

HUB CONTROLLER

The HUB controller has two endpoints, Endpoint0 and an Interrupt Endpoint. USB host can get the configuration information, the status of the HUB and also control the hub for enabling/disabling ports, turning power on and off, and other various controllable features.

HUB INTERFACE UNIT (HIU)

Like the FIU interface, the HUB Interface Unit interfaces with SIE on one side, but to the HUB command unit and the MCU interface on the other side. The MCU interface is provided for extra flexibility to allow MCU to handle certain commands like GET_DESCRIPTOR, and to be able to read hub and port status registers, and to handle any new or unimplemented commands in the HCU.

The HIU will have 8-byte bi-directional FIFOs for loading setup and out data from the host to the MCU, and loading in data from the MCU to the host. This block of logic will also handle the multiplexing of data from HCU and MCU to the SIE unit.

HUB COMMAND UNIT (HCU)

This block of logic executes the hub commands received from the host in a SETUP packet. Simple commands like SET FEATURE, CLEAR FEATURE or GET STATUS are executed in a single cycle and do not require multiple packets, these commands are implemented here in the hardware. Some commands like GET DESCRIPTORS take multiple packets to complete the transaction and these will be handled by the MCU. Also any command not implemented in the HCU will be passed on to MCU to handle.

HUB Command Unit executes the sub-set of USB commands for the hub, receive by the host, supported by the Compound Device in which the UsbCompM is implemented. It executes both the standard device class commands and the hub class commands. Not all the required commands and some of the optional commands are supported, following is the list of commands implemented. Rest of the required commands and other supported optional commands need to be implemented in the firmware in MCU.

Standard Requests:

SET_ADDRESS
GET_STATUS, device, interface, endpoints
GET_CONFIGURATION, device
SET_CONFIGURATION, device
SET_FEATURE, device, interface, endpoints
CLEAR_FEATURE, device, interface, endpoints
SET_INTERFACE
GET_INTERFACE

Hub Class Requests:

CLEAR_HUB_FEATURE
CLEAR_PORT_FEATURE, per port
GET_BUS_STATE, per port
GET_HUB_STATUS
GET_PORT_STATUS, per port
SET_HUB_FEATURE
SET_PORT_FEATURE, per port

Required Commands Implemented in MCU:

GET_DESCRIPTOR, device, configuration, interface, endpoints
GET_HUB_DESCRIPTOR

Optional Commands in MCU:

String descriptors
SET_DESCRIPTOR

FUNCTION INTERFACE UNIT (FIU)

The Function Interface consists of three endpoints; Endpoint0 and two programmable endpoints. Separate Endpoint0 is needed as hub Endpoint0 belongs to hub class and controlled by hub class driver, while Function Endpoint0 will most likely be controlled by the HID class driver. The other endpoints can be a bulk, isochronous or interrupt, with direction programmed as IN or OUT.

Function Interface Unit consists of Endpoint0 and two additional endpoints for the embedded function. The Endpoint0 logic consists of 8-byte bi-directional FIFO and all the control logic necessary to interface with the SIE on one side and with the MCU interface logic on the other side. The control logic keeps track of data toggle bit in a multiple packet transaction and resend of the data when the request is retried by the host. It handles the setting and clearing of the endpoint stall bit. The OUT/SETUP data from the FIFO is read by the MCU interface and data for the IN is loaded into the FIFO by MCU interface.

The two additional endpoints are programmable as IN or OUT Endpoint, and they can be interrupt, bulk or isochronous types. Each endpoint consists of 16-byte bi-directional FIFO used in only one direction, with direction programmed via a control bit in their respective CSR register. The data transfers between the MCU and the FIFOs are controlled by setting/clearing bits in the CSR. Interrupt may be generated on occurrence of some significant events and this interrupt can be disabled by the firmware.

MCU INTERFACE UNIT (MIU)

This block of logic will allow the MCU to interface to the HIU and FIU units. This block will handle the MCU timing, address decoding and data multiplexing from various units of the core to the MCU.

FRAME

The Frame timer runs at stable 12MHz clock and is reset when Start of Frame (SOF) is received from the host. This timer is used to generate End of Frame times (EOF1 and EOF2) needed by the RPTR block to detect babble and other error conditions. This timer is also used to generate Keep Alive signaling to the low speed downstream ports.

SUSPEND/RESUME

The suspend timer is used to detect inactivity on the upstream port. If no SOF is received for more than 3ms, hub enters a suspend state and SUSPEND signal is asserted. On detecting SUSPEND, STOP_CLK signal can be asserted by the MCU (or external hardware) to stop the clock in the UsbCompM.

When resume is detected by the upstream port control logic the SUSPEND signal is removed and resume signaling is sent downstream and suspend state is reset at the end of resume. Each downstream port control logic detects a remote wake-up causing resume to be send upstream and to all enable downstream ports. MCU can also handle remove wake-up by asserting RESUME_IN to the UsbCompM.

TRANSCEIVERS (XCVR)

The transceivers consist of a differential receiver, two single ended receivers and two drivers. There are five of these blocks, one for the upstream root port and rest for the four downstream ports. They are capable of transmitting and receiving data at 12 Mbps and 1.5 Mbps meeting the USB requirements.

CLOCK CIRCUITRY (CLK)

The clock circuitry can take in the crystal or oscillator input and generate 48 MHz and 12 MHz clock signal for rest of the logic blocks. These clocks may need to be stopped during the suspend mode depending on its current consumption.

MCU PROGRAMMING

MCU firmware need to support the Function Unit completely, as all the traffic, related to the embedded port will be relayed to the MCU by the UsbCompM core. In addition, MCU firmware also need to support certain Hub Unit related commands as pointed out in the previous section.

The Host commands supported by the Function Unit will depend on the device the firmware is implementing e.g. in monitor application HID class besides the required standard commands need to be supported. The USBCompM core presents number of registers to the MCU for controlling, monitoring and data transfers.

USB INTERRUPT HANDLING

USB Core Provides three distinct interrupts

- Hub Interrupt: Hub Endpoint, Embedded Function and Suspend/Resume
- Function Interrupt: Function Endpoints interrupt
- SOF: Start of Frame maker interrupt

HUB and Function interrupts have distinct enable and status registers. On initialization the interrupt enable registers must be set to "1", when MCU is ready to receive interrupts from the USB core (they may be set to "0" before setting them to "1" if MCU interrupts are edge sensitive). On getting an interrupt, MCU should read the corresponding status register and write the value just read to clear the interrupts. (If hub and function interrupts are ORed, both status registers should be read and written).

In the case of Embedded Function Interrupt, the Function Status Change register should be read and written back to clear its corresponding interrupt. SOF interrupt is generated every 1ms when hub frame timer is locked to the host. SOF is generated even when hub simulates the SOF.

USB REGISTER DESCRIPTION

GFI (GENERIC FUNCTION INTERFACE) REGISTERS

Some of the register in the HUB and the function unit are similar, specially pertaining to the endpoints. Description of these register will be presented only once here in the Generic Function Interface Register section to avoid duplication. Tables 11-1 to 11-3 lists all the USB registers.

Table 11-1. Function Register Description

Register Name	Address	R/W/C	Description
FADDR	BFH	R/W	Function USB address register
FEPSEL	BEH	R/W	Endpoint select register
FEPMAXP	BDH	R/W	Endpoint Max. packet size register
FEPCFG	BCH	R/W	Endpoint configuration register
FEPOUTCSR	BBH	R/W	Endpoint Out control/status register
FEPINCSR	BAH	R/W	Endpoint In control/status register
FEPBCNT	B9H	R	Function endpoint byte count register. Valid for out transfers.
FINTEN	B8H	R/W	Function interrupt enable register
FINTSTS	B7H	R/C	Function interrupt status register
FEP2BUF	B6H	R/W	Function endpoint 2 data buffer
FEP1BUF	B5H	R/W	Function endpoint 1 data buffer
FEP0BUF	B4H	R/W	Function endpoint 0 data buffer
FRAMEH	B1H	R	Upper bits of the Frame number
FRAMEL	B0H	R	Lower 8 bits of the Frame number. The Frame number register contains the frame number received with the last SOF.

Table 11-2. HUB/Function Common Register Description

Register Name	Address	R/W/C	Description
HINTEN	EBH	R/W	HUB interrupt enable register
HINTSTS	EAH	R/C	HUB interrupt status register
EMBFSTS	EDH	R/W	Embedded function status register, it is similar to embedded port status register but this one is for MCU, while embedded port register is for the host.
EMBFCSTS	ECH	R/C	Embedded function change status register, similar to embedded port register. Refer to explanation above

Table 11-3. Hub Register Description

Register Name	Address	R/W/C	Description
PXSTS3	FFH	R	Port status change MSB
PXSTS2	FEH	R	Port status change LSB
PXSTS1	FDH	R	Port status MSB
PXSTS0	FCH	R	Port status LSB
PORTSEL	FBH	R/W	Port select, selects port for status transfer = 1 downstream port 1 = 2 downstream port 2 = 3 downstream port 3 = 4 downstream port 4 = 5 embedded port
HEP0CSR	FAH	R/W	HUB Endpoint0 control/status register
HEP0BUF	F9H	R/W	HUB Endpoint0 data buffer
HEP0BCNT	F8H	R	HUB Endpoint0 byte count register, indicates bytes present in the OUT FIFO
HSTS3	F7H	R	HUB change status register MSB
HSTS2	F6H	R	HUB change status register LSB
HSTS1	F5H	R	HUB status register MSB
HSTS0	F4H	R	HUB status register LSB
HEP0STS1	F3H	R	HUB Endpoint 0 status MSB
HEP0STS0	F2H	R	HUB Endpoint 0 status LSB
HEP1STS1	F1H	R	HUB Endpoint 1 status MSB
HEP1STS0	F0H	R	HUB Endpoint 1 status LSB
DSTS1	EFH	R	Device status MSB
DSTS0	EEH	R	Device status LSB
HINTFSTS1	E7H	R	HUB interface status register
HINTFSTS0	E6H	R	HUB interface status register
PBUSSTE.	E5H	R	Port bus state, ports selected by PORTSEL register Bit0 = VMIN Bit1 = VPIN Bit7–Bit2 = 0
SUSRECON	E4H	R/C	Suspend/resume control register
MISCCON	E3H	R/W	Miscellaneous control register
STSCHGEP	E2H	R	Status change endpoint

FUNCTION ADDRESS REGISTER (FADDR)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	0	0	0	0	0	0	0
Read/Write	–	R/W	R/W	R/W	R/W	R/W	R/W	R/W

This register holds the USB address assigned by the host computer. FADDR is located at address BFH. This register is cleared by the core when port reset is received from the host for the embedded port or when USB_RESET has been received.

Bit7 Not used

Bit6–0 **FADDR:** MCU updates this register once it decodes a SET_ADDRESS command. MCU must write this register before it clears OUT_PKT_RDY (bit 0) and sets DATA_END (bit 4) in the EP0CSR register. This register is enabled for address comparison after the "status" phase of the SET_ADDRESS control transfer. This is so that the status IN packet which will still have "0" address can be recognized for this embedded function by the hardware in the SIU.

ENDPOINT SELECT REGISTER (FEPSEL)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	—	—	—	R/W	R/W

This register allows the MCU to select between Endpoint 0, 1, or 2 registers (CSR, BCNT, CFG and MAXP). FEPSEL is located at address BEH.

Bit 7–2 Not used

Bit 1–0 **FUNC_EP_SEL**: Endpoint 0, 1, and 2 register (CSR, BCNT, CFG, MAXP) share the same address space. To select between them, Endpoint Select register is provided and MCU can load the endpoint number 0, 1, or 2 into this register, before accessing any endpoint specific register. The buffer data is available for each endpoint at unique addresses and are independent of the 6 bit 0 and bit 1.

FUNCTION ENDPOINT CONFIGURATION REGISTER (FEPCFG)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (H/W)	R	R	R	R	—	—	—	—
Read/Write (MCU)	R/W	R/W	R/W	R/W	—	—	—	—

Function Endpoint Configuration register is used to configure Endpoint 1 and 2. Endpoint 0 is always a control endpoint. The register address is located at BCH. The can be read by hardware and read/write by MCU.

Bit 7 **DIRECTION:** This bit indicates the direction of the transfer between the host and the endpoint.
 1 = IN, endpoint to host
 0 = OUT, host to endpoint

Bit 6 **OUT_ISO:** If the Endpoint is set to OUT then:
 1 = ISO
 0 = Interrupt or Bulk

Bit 5 **IN_ISO:** If the Endpoint is set to IN then:
 1 = ISO
 0 = Interrupt or Bulk

Bit 4 **ISO_UPDATE:** This bit is used only in ISO mode for IN Endpoints. When this bit is set, a packet written to IN Endpoint is sent to the host only after SOF is received i.e. a packet loaded during frame N is sent to USB in frame N + 1. This ensures a complete frame for the next packet to be loaded.

Bit 3-0 Not used

FUNCTION ENDPOINT MAX PACKET REGISTER (FEPMAXP)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	1	0	0	0	0
Read/Write (H/W)	–	–	–	R	R	R	R	R
Read/Write (MCU)	–	–	–	R/W	R/W	R/W	R/W	R/W

This register is used to indicate the size of the Endpoint 1 and 2. FEPMAXP is located at address BDH.

The maximum size of the buffer for Endpoint1 and 2 is 16 bytes, if the max packet size is chosen to be 8 or less, two packets may be loaded to the buffer. This is recommended for an ISO endpoint, since it needs a constant bandwidth.

Bit 7-5 Not used

Bit 4-0 **MAX_PACKET_SIZE**: Maximum packet size for this endpoint.

ENDPOINT0 CSR REGISTER (EP0CSR)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	1	0	0	0	0
Read/Write (H/W)	R	R	W	R/C	W	R/C	R/C	W
Read/Write (MCU)	W	W	R/(C) ⁽²⁾	R/W	R/C	R/W	R/W	R/(C) ⁽²⁾

NOTES:

1. R/C The bit is cleared after being read
2. The bit is cleared by MCU by writing to another bit.

When Endpoint Select register (BEH) is equal to zero, Endpoint OUT CSR (BBH) and Endpoint IN CSR (BAH) selects the same Endpoint0 CSR. EP0CSR is located at FAH and is read/write addressable. Reading or writing to either address accesses the same register.

- Bit 7 **CLR_SETUP_END:** MCU writes "1" to this bit to clear SETUP_END bit (bit 5). This bit is not "sticky" and can be set in conjunction with other bits.
- Bit 6 **CLR_OUTPKT_RDY:** MCU writes "1" to this bit to clear OUT_PKT_RDY bit (bit 0). This bit is not "sticky" and can be set in conjunction with other bits.
- Bit 5 **SETUP_END:** The GFI sets this bit if the host ends the DATA phase before the specified number of bytes (the amount specified during SETUP phase), are transferred, or if the host comes back with a new control transfer before the current one is completed. Once the microcontroller sees this bit set, it should end the SETUP phase, stop loading/unloading the FIFO. Microcontroller does not set DATA_END in this case. The GFI, before setting this bit, flushes the FIFO, and prevents microcontroller's access to FIFO.
- Bit 4 **DATA_END:** During the DATA phase of a control transfer, after the microcontroller has finished loading/unloading the exact number of bytes as specified in the SETUP phase, this bit is set. This bit is cleared by GFI. If the transfer was to the host and last data packet to be transferred is equal to the max packet size, then setting of DATA_END should be postponed till IN_PKT_RDY is cleared. This will force a ZERO length data packet transfer from the core indicating to the host this packet is the last of the transfer.
- Bit 3 **STALL_SENT:** When the hardware decodes an illegal sequence from the host, it may send a STALL on its own to the USB host. This bit is set to inform the MCU that such an event has happened. This is informational only and does not cause an interrupt, and it needs to be cleared by the MCU after it has seen it.
- Bit 2 **FORCE_STALL:** When an illegal or unsupported command is decoded by the firmware it needs to set this bit. When set, this bit causes the hardware to return a STALL handshake to the host and is reset by the hardware when handshake has been sent. This bit should not be set when host does a SET_FEATURE STALL, as this will cause all transfers to/from endpoint0 to return STALL. This behavior is different for other endpoints.
- Bit 1 **IN_PKT_RDY:** The MCU, after filling the FIFO with IN data, sets this bit. MCU should wait for this bit to be cleared by the GFI before loading next IN token. If the function receives a valid IN token, while IN_PKT_RDY is not set by the MCU then the Endpoint 0 state machine issues a NAK handshake.
- Bit 0 **OUT_PKT_RDY:** The GFI sets this bits, whenever it has a valid token packet in the Endpoint0 FIFO. The MCU unloads the FIFO when this bit is set and clears this bit. If it is the SETUP phase, then the MCU also decodes the SETUP token, and checks to see if it is a valid command and, then clears this bit by doing CLR_EP0_OUTPKTRDY. At the time of clearing this bit, the MCU will also set FORCE_STALL if it is a invalid command, and DATA END if the length of data transfer during data phase is zero (no DATA phase, viz, SET_ADDRESS).

FUNCTION ENDPOINT OUT CSR (FEPOUTCSSR)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (H/W)	R	R/C	W	W	–	R/C	W	W
Read/Write (MCU)	W	W	R/C	R/C	–	R/W	R/C	R/C ⁽²⁾

NOTES:

1. R/C is the bit which is cleared after being read.
2. The bit is cleared by MCU by writing to another bit.

For endpoints other than 0, separate IN and OUT registers are available and micro-code should read the register the endpoint has been programmed for.

- Bit 7 **CLEAR_DATA_TOGGLE:** Writing a "1" to this bit will clear DATA 1/DATA 0 toggle bit. This should be done after a clear ENDPOINT_STALL has been received from the host. This bit is not sticky, in other words, it can be set in conjunction with other bits.
- Bit 6 **FLUSH_FIFO:** Setting this bit to "1" will flush the FIFO. Hardware will flush the FIFO if a OUT data transfer was not already in progress and it is cleared when flush is done.
- Bit 5 **DATA_ERR:** For ISO endpoint, hardware sets OUT_PKT_RDY even if the core has a CRC/bit stuffing error. But DATA_ERR bit is also set in this case. If the micro-code is capable of error recovery it can unload the packet, else it can flush the FIFO, which will clear out the FIFO and reset OUT_PKT_RDY.
- Bit 4 **SENT_STALL:** Hardware sets this bit when OUT transaction ended with STALL handshake.
This happens when:
1. Host sends more than MAXP data.
 2. FORCE_STALL is set.
- Bit 3 Not used
- Bit 2 **FORCE_STALL:** This bit is set by the MCU, when this bit is set, the function controller issues a STALL handshake to the host. This bit may be set by the MCU for any fault condition within the function or when host does a SET_FEATURE(ENDPOINT_STALL). It is cleared by the MCU when it receives a CLEAR_FEATURE(ENDPOINT_STALL) command from the host.
- Bit 1 **FIFO_ERROR:** This bit indicates if there was FIFO over-run or under-run in the previous transaction, and is used when the endpoint is configured as an ISO. If the direction was OUT and FIFO was not empty when data was received from the host, this bit will be set. If the direction was IN and FIFO was empty (IN_PKT_RDY, not set) and an IN transaction was received from the host, this bit will be set. This bit is informational only and no interrupt is generated. It can be cleared by writing a "0" to this bit.
- Bit 0 **OUT_PKT_RDY:** The GFI sets this bits, whenever it has a valid token packet in the endpoint1 FIFO. The MCU seeing this bit set, unloads the FIFO and clears this bit by writing "1" to this bit. At the time of clearing this bit, the MCU should also set FORCE_STALL if stall condition exists.

FUNCTION ENDPOINT IN CSR (FEPINCSR)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (H/W)	R	R/C	R/C ⁽²⁾	–	–	R/C	W	R/C ⁽²⁾
Read/Write (MCU)	W	W	R	–	–	R/W	R/C	R/W

NOTES:

1. R/C is the bit which is cleared after being read.
2. The bit is cleared by MCU by writing to another bit.

For Endpoint other than 0, separate in and out register are available and micro-code should read the register the endpoint has been programmed for. This register is located at address BAH and is read/write addressable.

- Bit 7 **CLR_DATA_TOGGLE:** Write a "1" to this bit to clear the DATA 1/DATA 0 toggle bit. This should be done after a clear ENDPOINT_STALL has been received from the Host. This bit is not sticky, it can be set in conjunction with other bits.
- Bit 6 **FLUSH_FIFO:** Setting this bit to a "1" will flush the FIFO. HW will flush the FIFO if a OUT data transfer was not already in progress and it is cleared when flush is done.
- Bit 5 **IN_PKT_RDY2:** When MCU writes a "1" to bit0, this bit is always gets set and is cleared by the hardware when all the packets have been transferred to the Host.
- Bit 4–3 Not used
- Bit 2 **FORCE_STALL:** This bit is set by MCU, whenever this bit is set, the function controller issues a STALL handshake to the host. This bit may be set by the MCU for any fault condition within the function or when host does a SET_FEATURE(ENDPOINT_STALL). It is cleared by MCU when it receives a CLEAR_FEATURE(ENDPOINT_STALL) command from the host.
- Bit 1 **FIFO_ERROR:** This bit indicates if there was a FIFO over run or under run in the previous transaction, and is used when the endpoint is configured as an ISO. If the direction was OUT and FIFO was not empty when data was received from the host, this bit will be set. If the direction was IN and FIFO was empty (IN_PKT_RDY not set) and an IN transaction was received from the host, this bit will be set.
- Bit 0 **IN_PKT_RDY:** MCU sets this bit after filling FIFO with a IN data. MCU should wait for this bit to be cleared by the GFI before loading next IN token. If the function receives a valid IN token, while IN_PKT_RDY is not set by the MCU then the endpoint state machine issues a NAK handshake.

In case where the max packet size is equal to or less than half the FIFO buffer size, this bit may not set even if MCU writes a "1". This is so that if MCU desires, it can load another packet to the FIFO, by checking this bit to be a "0" after the write. When second packet is loaded while the first packet is still in the FIFO, this bit will get set.

FUNCTION ENDPOINT WRITE COUNT REGISTER (FEPBCNT)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	R	R	R	R	R

EP0BCNT register has the number of valid bytes in Endpoint 0 FIFO. It is located at address B9H.

Bit 7–5 Not used

Bit 4–0 **Byte Count**: Once the MCU detects OUT_PKT_RDY set in CSR, then it can read this register to find out the number of bytes to be read from Endpoint 0 FIFO.

FRAME NUMBER REGISTER HIGH and LOW (FRAMEH and FRAMEL)

(HIGH) Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

(LOW) Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Frame Number register High (B1H) and Low (B0H) contains the frame number from the SOF packet

Bit 15–12 Not used

Bit 11 **ERR:** If a CRC error is detected, ERR flag is set so that FW can ignore the frame number and use its internal value or wait for the next SOF if this happens on SYNCH_FRAME time.

Bit 10–0 **CURRENT FRAME NUMBER:** On detection of SOF from the Host, this register is updated with the frame number received with the SOF packet. After receiving a SYNCH_FRAME request from the host, firmware should start monitoring SOF, and read this register on getting SOF interrupt and send the value (2 bytes) back to the host. And from then on it can keep track of the frame number itself, incrementing on SOF.

INTERRUPT ENABLE REGISTER (FINTEN)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	–	–	0	0	0
Read/Write	–	–	–	–	–	R/W	R/W	R/W

This register serves as interrupt mask register. By default upon reset, all the interrupt are disabled. If an interrupt is being serviced, firmware may want to mask the interrupt(s) by masking the corresponding bit(s) or when certain interrupt status bits are going to be polled. This register is located at address B8H.

Bit 7–3 Not used

Bit 2 **ENDPNT2_INTEN:**
1 = enable interrupt
0 = disable interrupt

Bit 1 **ENDPNT1_INTEN:**
1 = enable interrupt
0 = disable interrupt

Bit 0 **ENDPNT0_INTEN:**
1 = enable interrupt
0 = disable interrupt

INTERRUPT STATUS REGISTER (FINTSTS)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (H/W)	—	—	—	—	—	W	W	W
Read/Write (MCU)	—	—	—	—	—	R/C	R/C	R/C

This register is used to indicate what condition caused a interrupt to the MCU. Refer to Interrupt Handling section on details for clearing the interrupts. FINTSTS is located in address B7H.

Bit 7–3 Not used

Bit 2 **ENDPNT2:** This bit is set by GFI, Endpoint 2 needs to be serviced.

- IN_PKT_RDY gets cleared for IN
- OUT_PKT_RDY gets set for OUT

Bit 1 **ENDPNT1:** This bit is set by GFI, when Endpoint 1 needs to be serviced.

- IN_PKT_RDY gets cleared for IN
- OUT_PKT_RDY gets set for OUT

Bit 0 **ENDPOINT0 (CONTROL):** This bit is set by GFI, upon following condition, when Endpoint 0 needs to be serviced. It is set under any one of the following conditions:

- OUT_PKT_RDY is set.
- IN_PKT_RDY gets cleared.
- DATA_END gets cleared.
- SETUP_END gets set.

HUB INTERRUPT ENABLE REGISTER (HINTEN)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	–	–	–	–

This register serves as interrupt mask register. By default upon reset, all the interrupts are disabled. If an interrupt is being serviced firmware may want to mask the interrupt(s) by masking the corresponding bit(s) or when certain interrupt status bits are going to be polled. HINTEN is located in address EBH.

Bit 7 **HUB:**
 1 = enable interrupt
 0 = disable interrupt

Bit 6 **SUSP_RESM:**
 1 = enable interrupt
 0 = disable interrupt

Bit 5 **CONN_LOST:**
 1 = enable interrupt
 0 = disable interrupt

Bit 4 **EMB_FUNC:**
 1 = enable interrupt
 0 = disable interrupt

Bit 3–0 Not used

HUB INTERRUPT STATUS REGISTER (HINTSTS)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (H/W)	W	W	W	W	—	—	—	—
Read/Write (MCU)	R/C	R/C	R/C	R/C	—	—	—	—

HINTSTS is located at address EAH and is read only. This register is used to indicate what condition caused a interrupt to the MCU. Refer to Interrupt Handling section on details for clearing the interrupts

- Bit 7 **HUB:** This bit is set when hub Endpoint0 need serving. This bit is similar to ENPOINT 0 interrupt explained earlier for the function.
- Bit 6 **SUSP_RESM:** This bit is set when the hub goes into the suspend mode or coming out of it. This happens before clocks are stopped and when clocks have resumed for the resume case. This is so the MCU may take appropriate action.
- Bit 5 **CONN_LOST:** This bit is set when CONN in Function status register is a "0" and resets when CONN is set to "1". This bit can also be used as an indicator for hub reset.
- Bit 4 **EMB_FUNCTION_STATUS_CHANGE:** This bit is set when a bit in the Function Status Change register is set to "1".
- Bit 3–0 Not used

HUB EP0 CSR REGISTER (HEP0CSR)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	1	0	0	0	0
Read/Write (H/W)	R	R	W	R/C	W	R/C	R/C	W
Read/Write (MCU)	W	W	R/C ⁽²⁾	R/W	R/C	R/W	R/W	R/C ⁽²⁾

NOTES: 1. R/C The bit is cleared after being read.
2. The bit is cleared by MCU by writing to another bit.

HEP0CSR is located at address FAH and is read/write addressable. Please refer to the EP0CSR register, the bit definition and the meaning are the same for both HUB and function.

HUB EP0 BUFFER (HEP0BUF)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HUB Endpoint0 buffer is located at address F9H. HEP0BUF is 8-byte deep and can be read or written to by the MCU, depending on the phase of the control transaction. Writes to the buffer are permitted only in control transaction while in progress. Writes to the buffer are not permitted while preparing to load SETUP packet from the host and when in an OUT phase of the control transfer.

Data can be read from the EP0 buffer when an OUT_PKT_RDY has been set. When a read is attempted while OUT_PKT_RDY is not set or SETUP_END condition is set, invalid or zero data will be returned. MCU should read byte count provided in the HUB_EP0_BCNT register.

HUB EP0 BCNT REGISTER (HEP0BCNT)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

HEP0BCNT is located at address F8H. Lower four bits of this register consists of the byte count available in the EP0 buffer. Upper bits are returned zero.

EMBEDDED FUNCTION STATUS REGISTER (EMBFSTS)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (H/W)	–	–	–	W	–	W	W	R/C
Read/Write (MCU)	–	–	–	R	–	R	R	R/W

This register of relevant bits of status register for the embedded function set by the host via hub controller, except for CONN which is set by the MCU. EMBFSTS is located at address EDH and is read/write addressable.

Bit 7–5 Not used

Bit 4 **RST**: Set by the host to selectively reset the function. MCU can reset its registers and go to Idle state on assertion of this bit. This is cleared by the hub when RESET bit in the embedded port register has been cleared.

Bit 3 Not used

Bit 2 **SUSP**: Set by the host to selectively suspend the embedded function. This bit is same as one in the port status register sent to the host by the HUB. MCU is supposed to go into low power mode on assertion of this bit. It can be reset by the host or when MCU sends remote resume.

Bit 1 **ENB**: Set by the host, this is for information only no action is required by the MCU.
1 = function is enabled
0 = function is disabled

Bit 0 **CONN**: Set by the MCU to indicate to the hub controller that embedded function is present and ready. RESET by the hardware when detecting USB RESET from the host or power on RESET.

EMBEDDED FUNCTION STATUS CHANGE REGISTER (EMBFCSTS)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (H/W)	–	–	–	W	–	W	–	–
Read/Write (MCU)	–	–	–	R/C	–	R/C	–	–

EMBFCSTS is located at address ECH. This register consists of relevant bits of status change register for the embedded function. These bits are set when corresponding bit in status register changes and an interrupt is set to the MCU. These bits are cleared by writing a "1" to the corresponding bit position.

Bit 7–5 Not used

Bit 4 **c_RST**: This bit is set on the assertion of RST bit in the status register. Interrupt is set. Clearing this bit removes the interrupt.

Bit 3 Not used

Bit 2 **c_SUSP**: Set to "1", when SUSP is set or reset in the status register. Interrupt is set. Clearing this bit removes the interrupt.

Bit 1-0 Not used

SUSPEND AND RESUME STATUS REGISTER (SUSRECON)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

SUSRECON is located at address E4H.

Bit 7–3 Not used

- Bit 2 **SUSPEND_OUT:** A distinct SUSPEND_OUT signal is also generated by the core when HUB actually enters low power suspend mode which is 2ms after HUB suspend has been set., which may be used to assert STOP_CLOCK signal by the MCU core. This is signal that is also readable in this register. SUSPEND_OUT may also be wired to another interrupt line in the MCU at a higher priority.
- Bit 1 **RESUME_IN:** This bit is set by MCU when it wants to initiate a remote resume. This bit is not sticky i.e. it's read as zero. This bit can be set when the embedded port has been suspended (susp bit in Emb. Func. Sts) or when the hub has been suspended (HUB Suspend), and MCU wants to resume. If this bit is set when HUB itself was suspended, global resume is carried out. Else if hub was awake, the embedded port status is changed to inform the host. And also Embedded Function status and Change Status register are updated interrupt issued to the MCU.
- Bit 0 **HUB_SUSPEND:** This bit is set when the HUB controller detects no activity on the root hub in excess of 3ms and the HUB is ready to go into low power suspend mode. The SUSP_RESUM interrupt is generated on setting of this bit, and this interrupt is ORed with other HUB interrupts. The MCU gets an early warning that HUB is going into suspend mode. When the hub is resumed, by root or a port, this is cleared and SUSP_RESUM interrupt is again set.

MISCELLANEOUS CONTROL REGISTER (MISCCON)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write (H/W)	—	—	—	—	—	—	R	R
Read/Write (MCU)	—	—	—	—	—	—	R/W	R/W

Bit 7–2 Not used

Bit 1 **HUB_CMD_XCP_EN**: Enable MCU handling of certain hub commands, like GET_DESCRIPTOR.
 1 = Command decoder will take exception, whenever GET_DESCRIPTOR or unknown request from the host is decoded.
 0 = Command decoder will not take exception, instead will execute the request or return the stall to the host.

Bit 0 **USB_RST_EN**: USB RESET Enable controls the generation of USB_RSTN
 1 = USB_RSTN is asserted when USB RESET is detected.
 0 = USB_RSTN remains high.

PORT SELECT REGISTER (PORTSEL)

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PORTSEL register is used to select the port for the reads of port status registers. This is a 3-bit register and upper values are zero. This register is located at address FBH and is read/write addressable.

1 = Downstream port 1
 2 = Downstream port 2
 3 = Downstream port 3
 4 = Downstream port 4
 5 = Embedded port

NOTES

12

ELECTRICAL DATA

OVERVIEW

In this section, the following KS86C6308/P6308 electrical characteristics are presented in tables and graphs:

- Absolute maximum ratings
- D.C. electrical characteristics
- Input/Output capacitance
- A.C. electrical characteristics
- Input timing for external interrupt (Ports 0, 2 and 4) DP0/GPIO, DM0/GPIO : GPIO Mode Only
- Input timing for RESET
- Oscillator characteristics
- Oscillation stabilization time
- Clock timing measurement points at X_{IN}
- Data retention supply voltage in Stop mode
- Stop mode release timing when initiated by a reset
- Stop mode release timing when initiated by an external interrupt
- Characteristic curves

Table 12-1. Absolute Maximum Ratings

(T_A = 25°C)

Parameter	Symbol	Conditions	Rating	Unit
Supply Voltage	V _{DD}	–	– 0.3 to + 6.5	V
Input Voltage	V _{IN}	All input ports	– 0.3 to V _{DD} + 0.3	V
Output Voltage	V _O	All output ports	– 0.3 to V _{DD} + 0.3	V
Output Current High	I _{OH}	One I/O pin active	– 18	mA
		All I/O pins active	– 60	
Output Current Low	I _{OL}	One I/O pin active	+ 30	mA
		Total pin current for ports 0, 1, 2, 4	+ 100	
		Total pin current for port 3	+ 100	
Operating Temperature	T _A	–	– 40 to + 85	°C
Storage Temperature	T _{STG}	–	– 65 to + 150	°C

Table 12-2. D.C. Electrical Characteristics

(T_A = -40 °C to +85 °C, V_{DD} = 4.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Operating Voltage	V _{DD}	f _{OSC} = 12 MHz	4.0	—	5.5	V
Input High Voltage	V _{IH1}	All input except V _{IH2}	0.8 V _{DD}	—	V _{DD}	V
	V _{IH2}	X _{IN}	V _{DD} - 0.5		V _{DD}	
Input Low Voltage	V _{IL1}	All input pins except V _{IL2}	—	—	0.2 V _{DD}	V
	V _{IL2}	X _{IN}			0.4	
Output High Voltage	V _{OH}	I _{OH} = -200 µA; All output ports except ports 0, 1, 2, DP's, DM's	V _{DD} - 1.0	—	—	V
Output Low Voltage	V _{OL}	I _{OL} = 1 mA All output ports except DP's, DM's	—	—	0.4	V
Input High Leakage Current	I _{LIH1} ⁽⁴⁾	V _{IN} = V _{DD} All inputs excepts I _{LIH2} , DP's, DM's	—	—	3	µA
	I _{LIH2} ⁽⁴⁾	V _{IN} = V _{DD} X _{IN} , X _{OUT} , RESET	—	—	20	µA
Input Low Leakage Current	I _{LIL1} ⁽⁴⁾	V _{IN} = 0 V All inputs excepts I _{LIL2} , DP's, DM's	—	—	-3	µA
	I _{LIL2} ⁽⁴⁾	V _{IN} = 0 V X _{IN} , X _{OUT} , RESET	—	—	-20	µA

Table 12-2. D.C. Electrical Characteristics (continued)

(T_A = -40 °C to +85 °C, V_{DD} = 4.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Output High Leakage Current	I _{LOH} ⁽¹⁾	V _{OUT} = V _{DD} All I/O pins and output pins except DP's and DM's	—	—	3	μA
Output Low Leakage Current	I _{LOL} ⁽¹⁾	V _{OUT} = 0 V All I/O pins and output pins except DP's and DM's	—	—	- 3	μA
Pull-up Resistors	R _L	V _{IN} = 0 V Ports 0, 1, 2, 4, Reset	25	50	100	kΩ
Supply Current	I _{DD1}	Normal operation mode : 12 MHz Crystal Oscillator	—		30	mA
	I _{DD2}	Idle mode; 12 MHz Crystal Oscillator			15	
	I _{DD3}	Stop mode: Oscillator stop			500	μA

NOTES:

1. Except X_{IN} and X_{OUT}.
2. Supply current does not include through internal pull-up resistors or external output current loads.
3. Figure 11-3 Transition Rise Timer (t_R), Fall Timer (t_F) parameter is guaranteed, but not tested.
3. When USB Mode Only in 4.20 V to 5.25 V, DP's and DP's satisfy the USB Specification version 1.0.

Table 12-3. Input/Output Capacitance

(T_A = -40 °C to +85 °C, V_{DD} = 0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input Capacitance	C _{IN}	f = 1 MHz; Unmeasured pins are connected to V _{SS}	—	—	10	pF
Output Capacitance	C _{OUT}					
I/O Capacitance	C _{IO}					

Table 12-4. A.C. Electrical Characteristics

(T_A = -40 °C to +85 °C, V_{DD} = 4.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Interrupt Input High, Low Width	t _{INTH} , t _{INTL}	P0, P2 and P4	—	200	—	ns
RESET Input Low Width	t _{RSL}	RESET	—	1000	—	

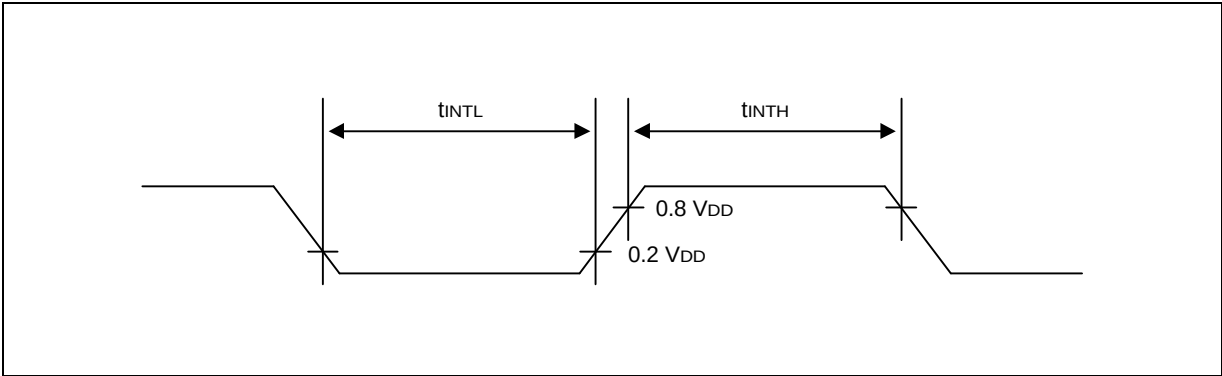


Figure 12-1. Input Timing Measurement Points (Ports 0, 2, and 4)

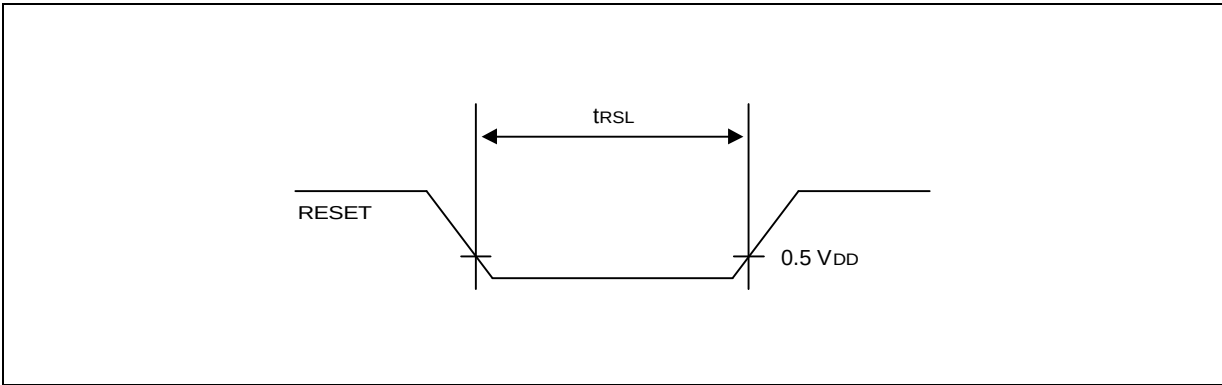


Figure 12-2. Input Timing for RESET

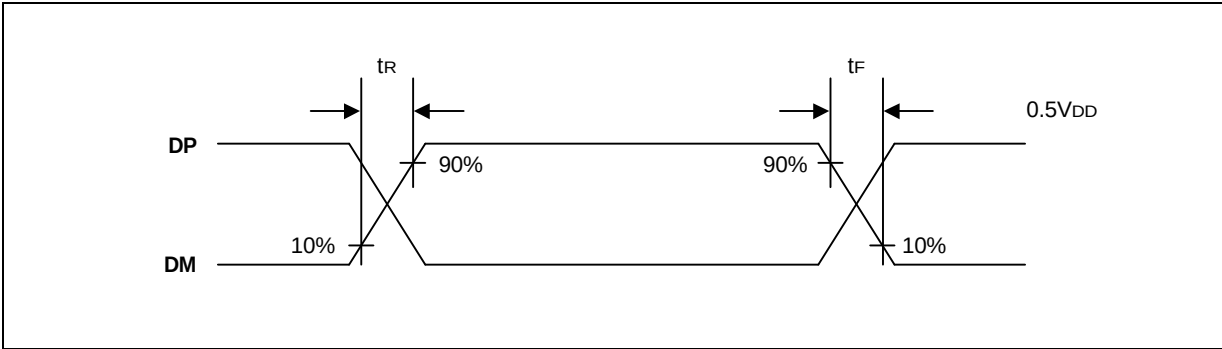


Figure 12-3. USB Data Signal Timing

Table 12-5. DPx, DMx Driver Characteristics, Full Speed Operation

Symbol	Parameter	Condition	Min	Max	Unit
t_R	Rise Time	$C_L = 50\text{pF}$	4	20	ns
t_F	Fall Time	$C_L = 50\text{pF}$	4	20	ns
t_{RFM}	t_R/t_F Matching	—	90	11	%

Table 12-6. DPx, DMx Driver Characteristics, Low Speed Operation

Symbol	Parameter	Condition	Min	Max	Unit
t_R	Rise Time	$C_L = 200\text{-}600\text{pF}$	75	300	ns
t_F	Fall Time	$C_L = 200\text{-}600\text{pF}$	75	300	ns
t_{RFM}	t_R/t_F Matching	—	80	125	%

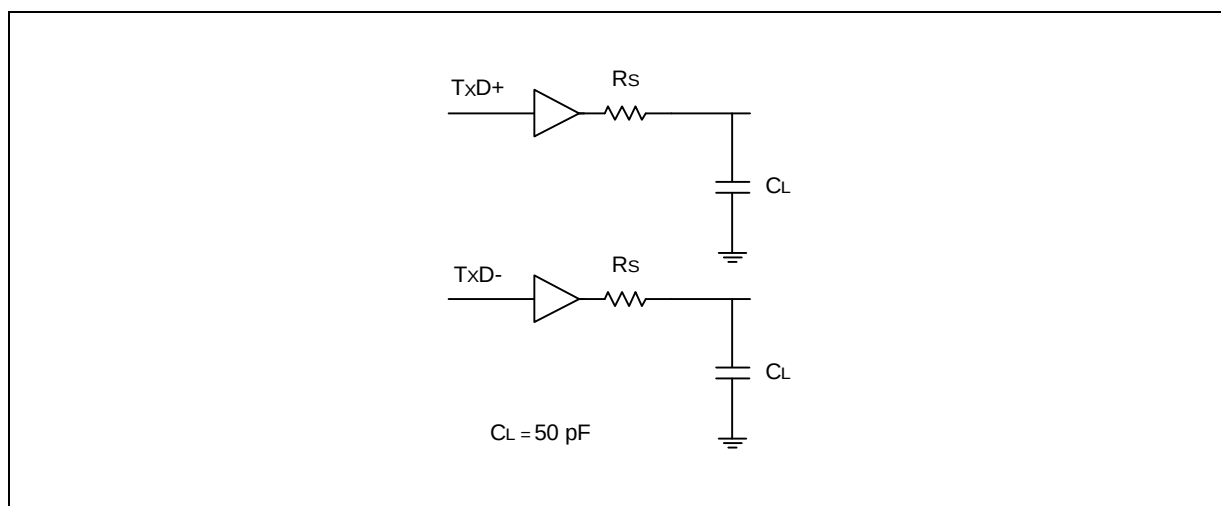


Figure 12-4. Full-Speed Load

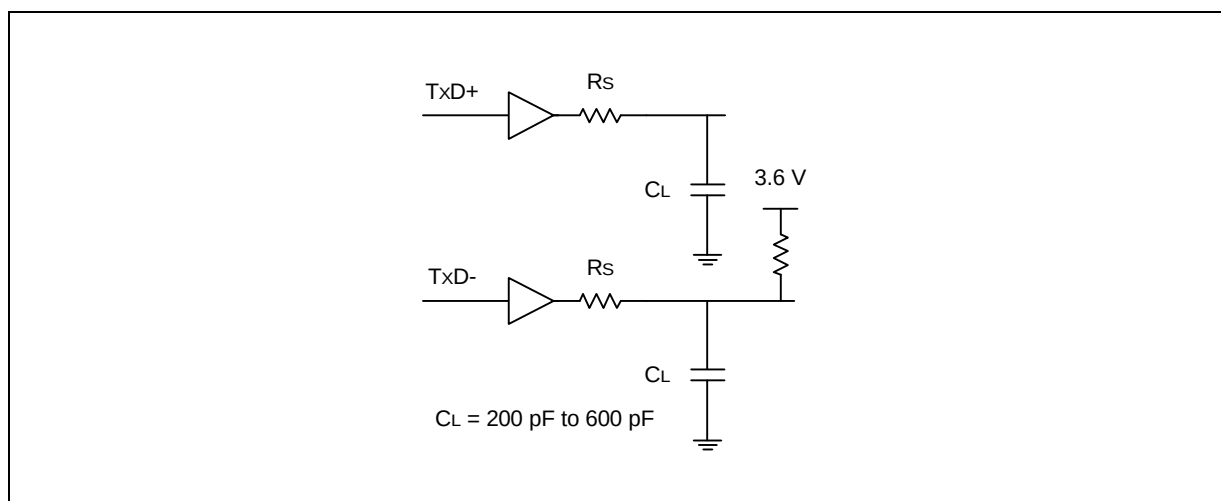


Figure 12-5. Low-Speed Load

Table 12-7. Oscillator Characteristics

(T_A = -40°C + 85°C)

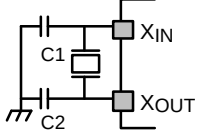
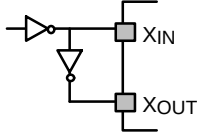
Oscillator	Circuit	Condition	Min	Typ	Max	Unit
Main crystal Main ceramic (f _{OSC})		V _{DD} = 4.0V to 5.5V	–	12	–	MHz
External clock		V _{DD} = 4.0V to 5.5V	–	12	–	

Table 12-8. Oscillation Stabilization Time

(T_A = -40°C + 85°C, V_{DD} = 4.0 V to 5.5 V)

Oscillator	Symbol	Condition	Min	Typ	Max	Unit
Crystal	–	V _{DD} = 4.0V to 5.5V	–	–	20	ms
Ceramic	–		–	–	10	
External	–	XIN input high & low level width	25	–	500	ns

NOTE: The oscillator stabilization wait time, t_{WAIT}, is determined by the setting in the basic timer control register, BTCON.

Table 12-9. Data Retention Supply Voltage in Stop Mode

(T_A = -40°C to +85°C)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data Retention Supply Voltage	V _{DDDR}	Stop mode	2.0	–	6	V
Data Retention Supply Current	I _{DDDR}	Stop mode; V _{DDDR} = 2.0 V	–	–	500	μA

13

MECHANICAL DATA

OVERVIEW

The KS86C6308/P6308 is available in a 64-pin SDIP package (Samsung: 64-SDIP-750) and a 64-pin QFP package (64-QFP-1420F). Package dimensions are shown in Figures 13-1 and 13-2.

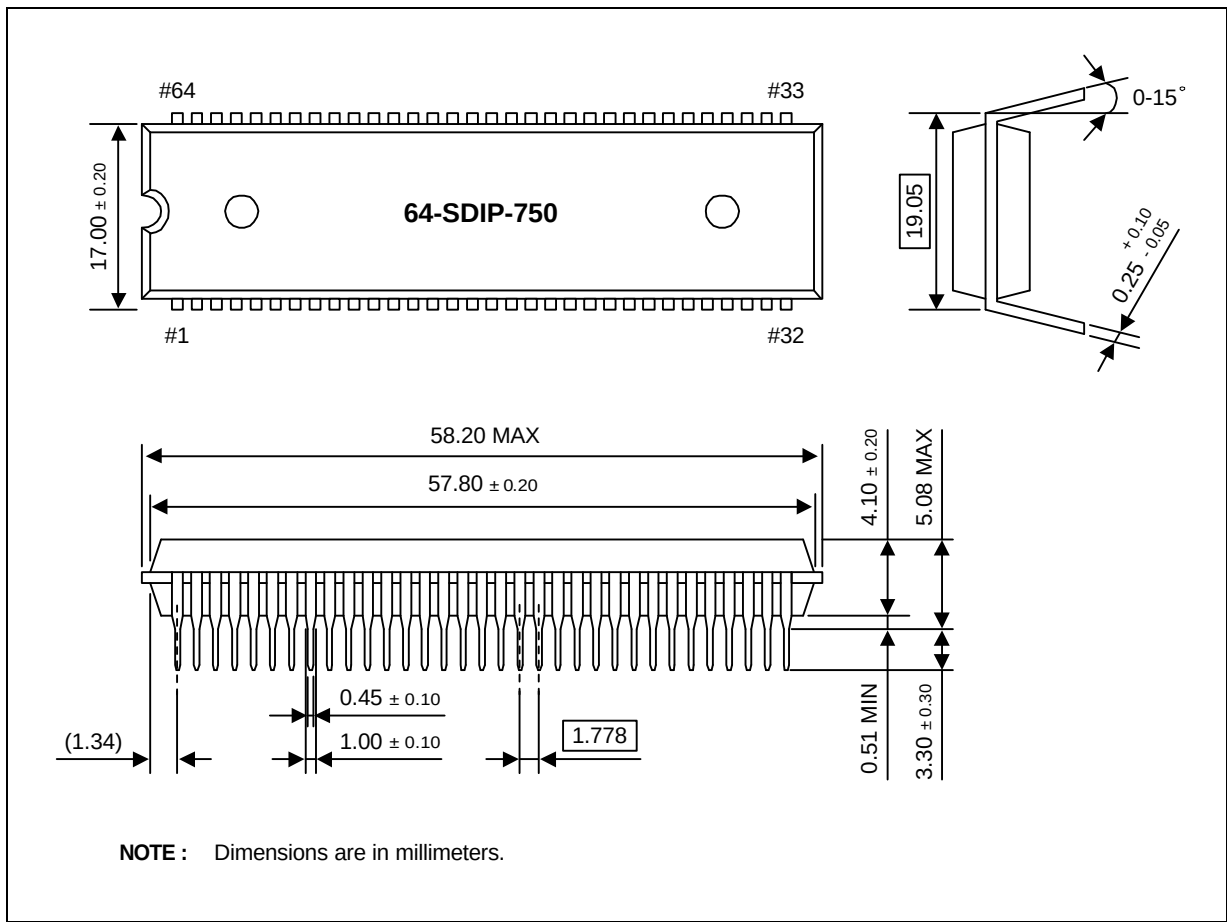
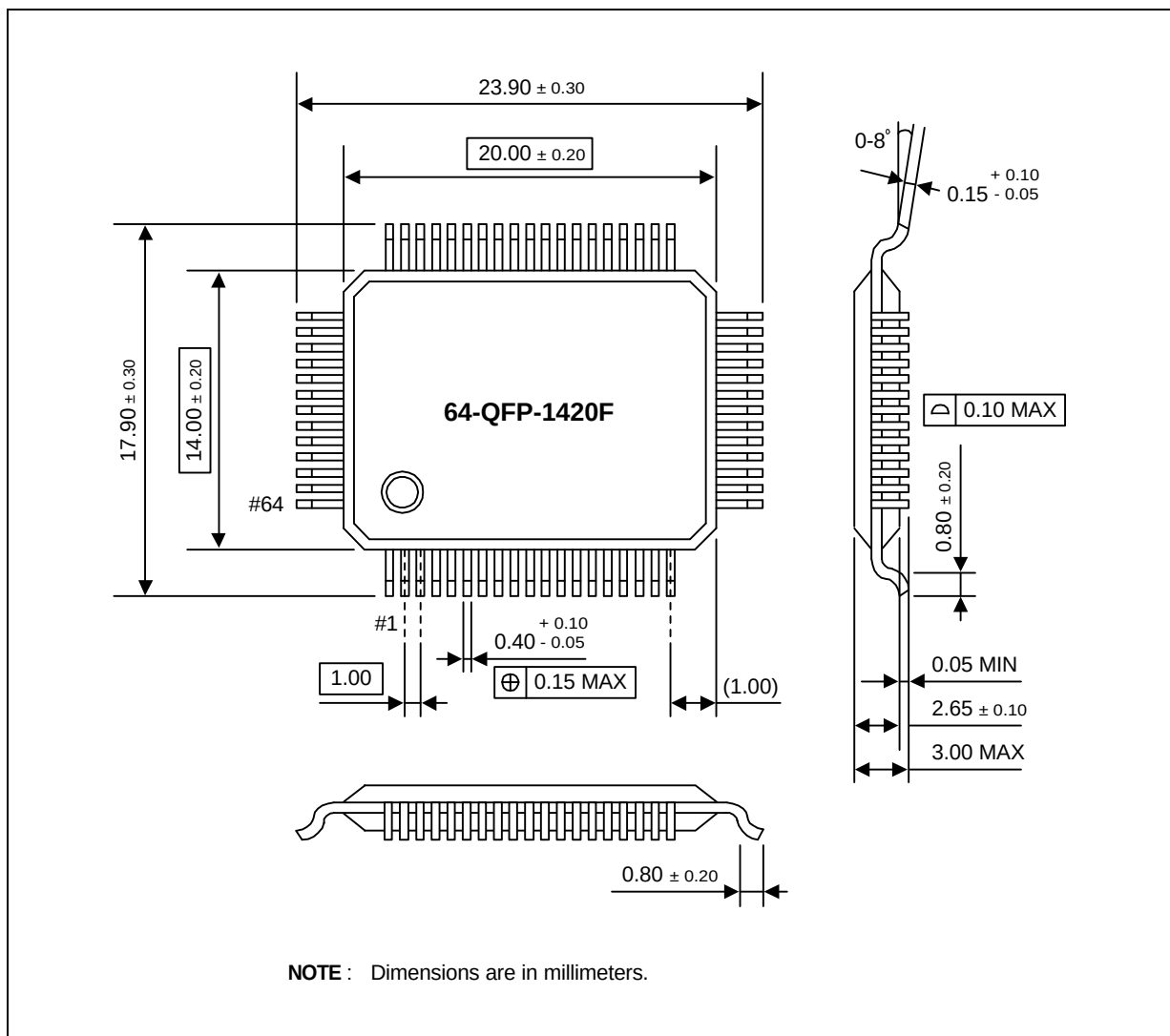


Figure 13-1. 64-Pin SDIP Package Mechanical Data (64-SDIP-750)



14

KS86P6308 OTP

OVERVIEW

The KS86P6308 single-chip CMOS microcontroller is the OTP (One Time Programmable) version of the KS86C6308 microcontroller. It has an on-chip OTP ROM instead of masked ROM. The EPROM is accessed by serial data format.

The KS86P6308 is fully compatible with the KS86C6308, both in function and in pin configuration. Because of its simple programming requirements, the KS86P6308 is ideal for use as an evaluation chip for the KS86C6308.

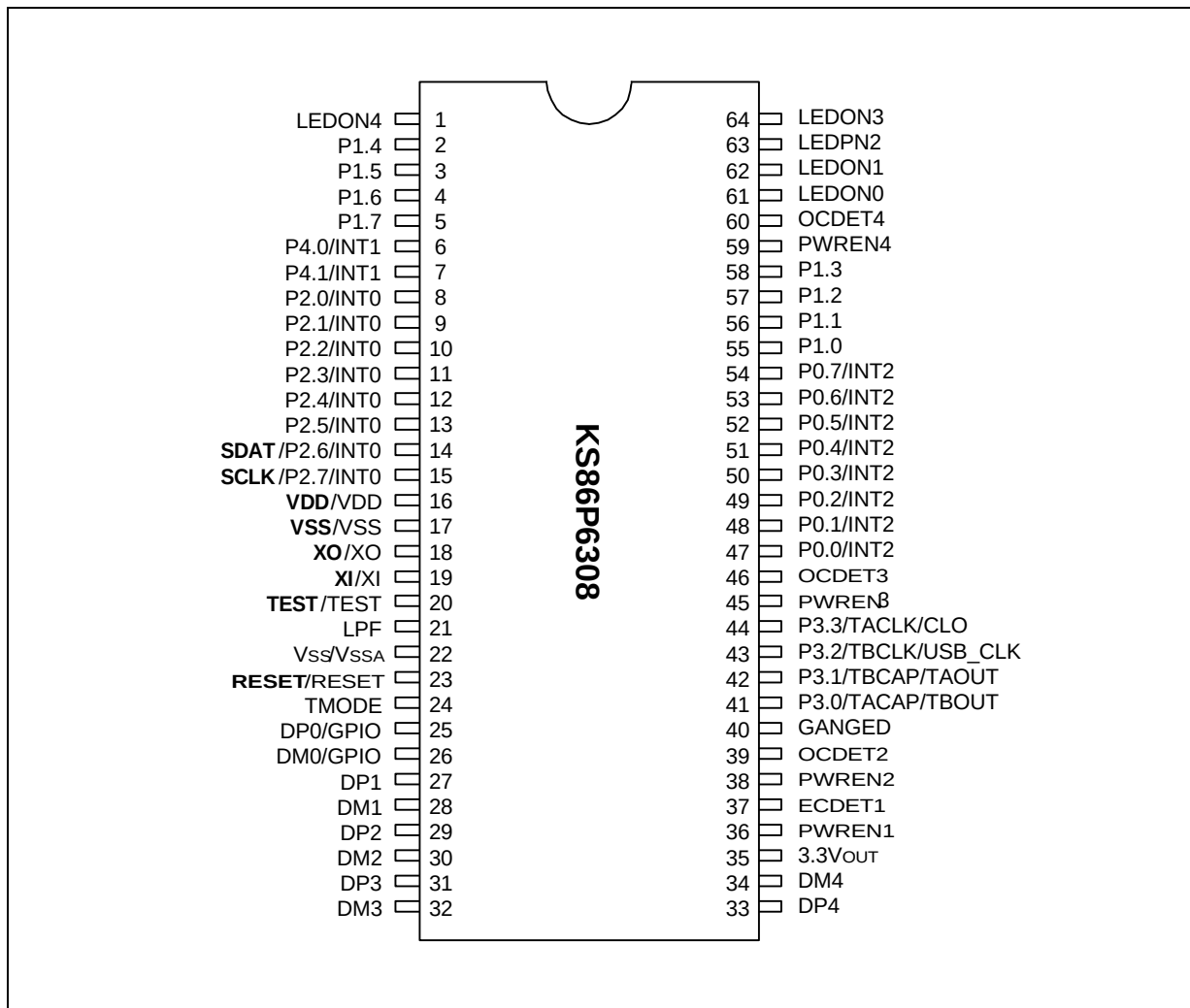


Figure 14-1. Pin Assignment Diagram (64-Pin SDIP Package)

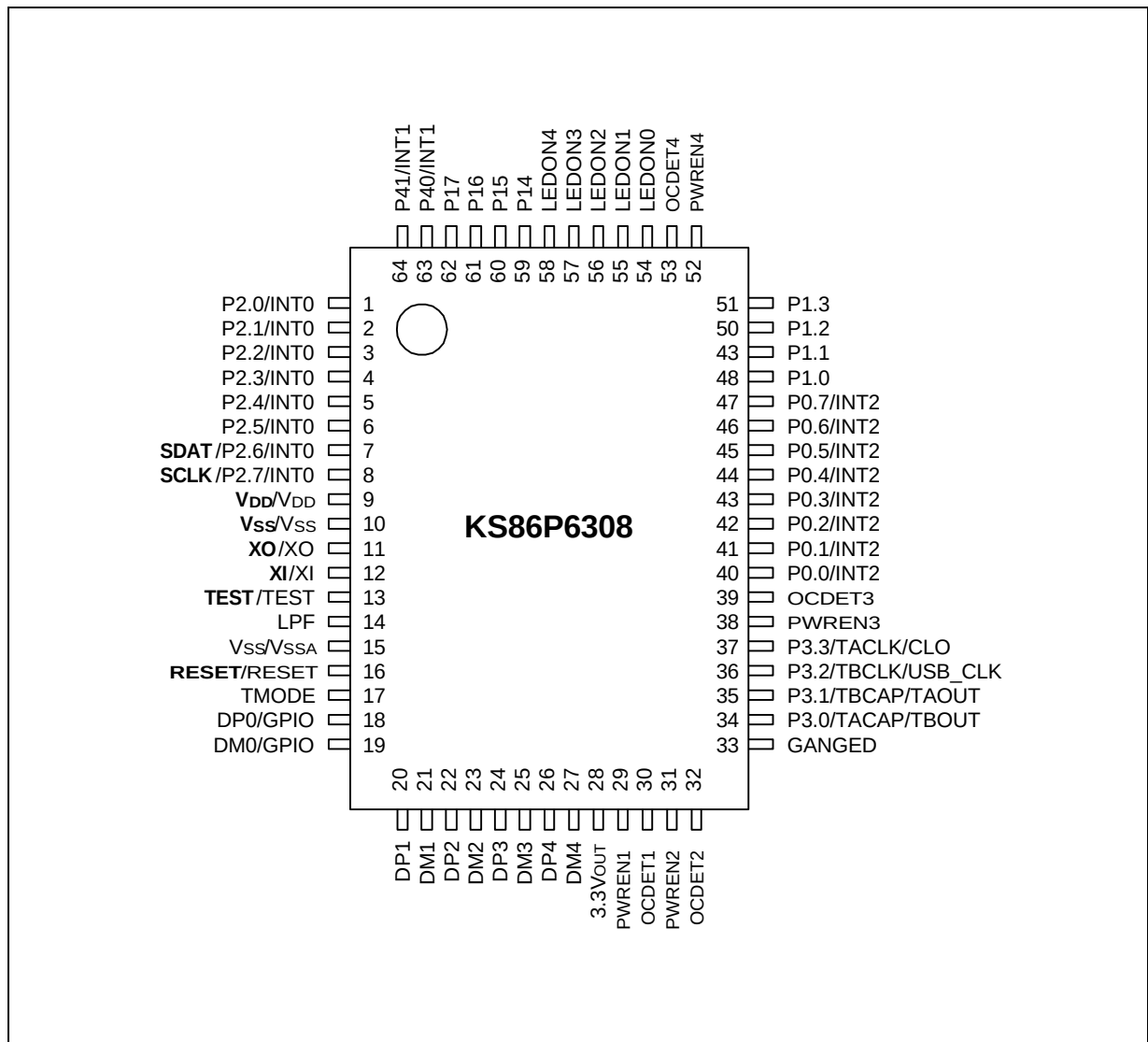


Figure 14-2. Pin Assignment Diagram (64-Pin QFP Package)

Table 14-1. Descriptions of Pins Used to Read/Write the EPROM

Main Chip Pin Name	During Programming			
	Pin Name	Pin No.	I/O	Function
P2.6	SDAT	9 ⁽³⁾	I/O	Serial Data Pin (Output when reading, Input when writing) Input and Push-pull Output Port can be assigned
P2.7	SCLK	10 ⁽⁴⁾	I/O	Serial Clock Pin (Input Only Pin)
TEST	TEST	15 ⁽⁹⁾	I	Chip Initialization and EPROM Cell Writing Power Supply Pin (Indicates OTP Mode Entering) When writing 12.5 V is applied and when reading.
RESET	RESET	18 ⁽¹²⁾	I	0 V: OTP write and test mode 5 V: Operating mode
V _{DD} / V _{SS}	V _{DD} / V _{SS}	11 ⁽⁵⁾ /12 ⁽⁶⁾	–	Logic Power Supply Pin.

NOTE: () means 64 QFP package.

Table 14-2. Comparison of KS86P6308 and KS86C308 Features

Characteristic	KS86P6308	KS86C6308
Program Memory	8-Kbyte EPROM	8-Kbyte mask ROM
Operating Voltage (V _{DD})	4.0 V to 5.25 V	4.0 V to 5.25 V
OTP Programming Mode	V _{DD} = 5 V, V _{PP} (RESET) = 12.5 V	
Pin Configuration	64 SDIP/64 QFP	64 SDIP/64 QFP
EPROM Programmability	User Program 1 time	Programmed at the factory

OPERATING MODE CHARACTERISTICS

When 12.5 V is supplied to the VPP (RESET) pin of the KS86P6308, the EPROM programming mode is entered. The operating mode (read, write, or read protection) is selected according to the input signals to the pins listed in Table 14-3 below.

Table 14-3. Operating Mode Selection Criteria

V _{DD}	V _{PP} (RESET)	REG/ MEM	ADDRESS (A15-A0)	R/W	MODE
5 V	5 V	0	0000H	1	EPROM read
	12.5 V	0	0000H	0	EPROM program
	12.5 V	0	0000H	1	EPROM verify
	12.5 V	1	0E3FH	0	EPROM read protection

NOTE: "0" means Low level; "1" means High level.

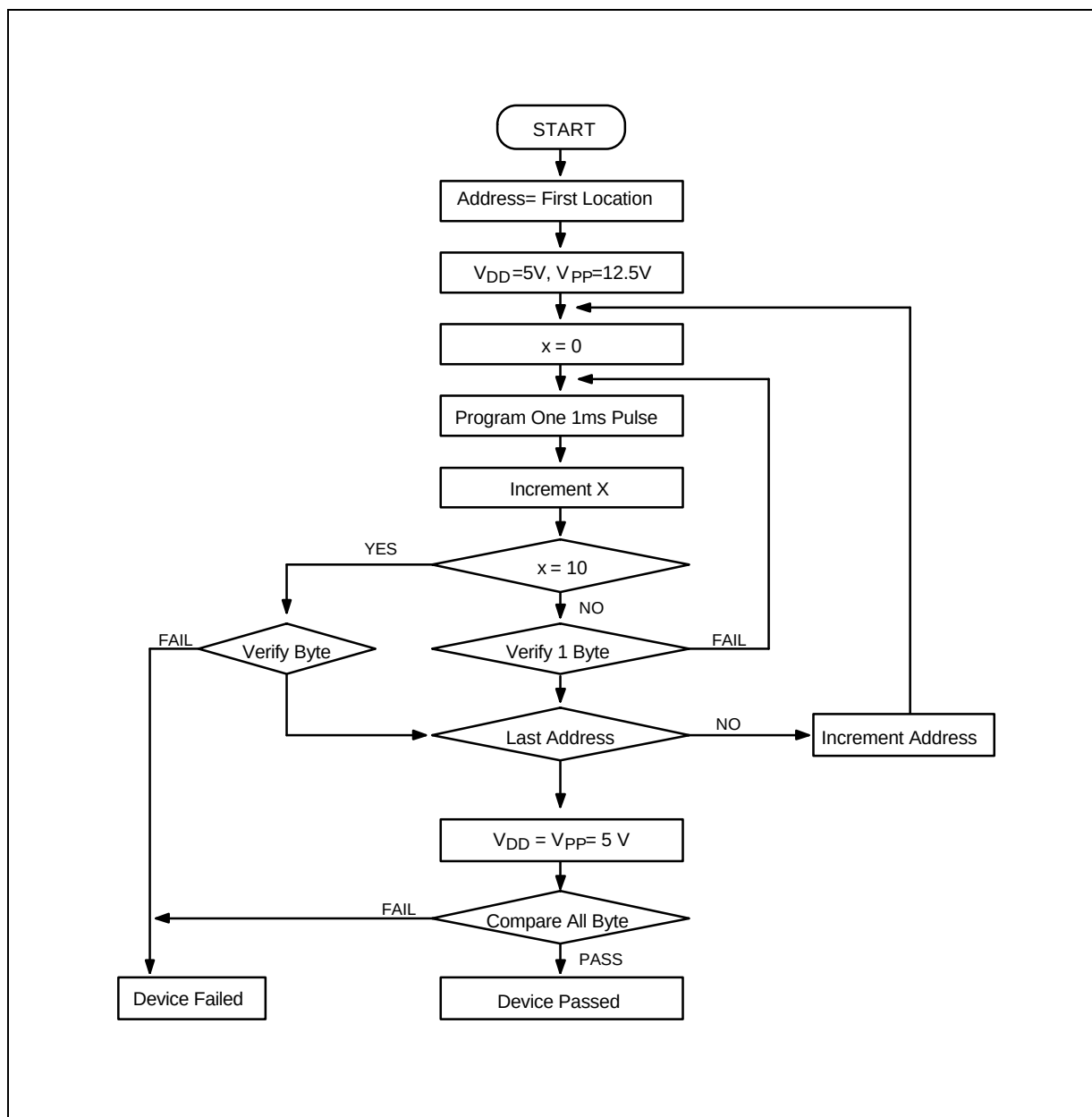


Figure 14-3. OTP Programming Algorithm

Table 14-4. D.C. Electrical Characteristics $(T_A = -40^{\circ}\text{C to } +85^{\circ}\text{C}, V_{DD} = 5.25\text{ V})$

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Supply Current (note)	I_{DD1}	Normal mode; 12 MHz crystal oscillator	–	–	30	mA
	I_{DD2}	Idle mode; 12 MHz CPU clock		–	15	
	I_{DD3}	Stop mode		–	500	μA

NOTE: Supply current does not include current drawn through internal pull-up resistors or external output current loads.

15

DEVELOPMENT TOOLS

OVERVIEW

Samsung provides a powerful and easy-to-use development support system in turnkey form. The development support system is configured with a host system, debugging tools, and support software. For the host system, any standard computer that operates with MS-DOS as its operating system can be used. One type of debugging tool including hardware and software is provided: the sophisticated and powerful in-circuit emulator, SMDS2+, for KS57, KS86, KS88 families of microcontrollers. The SMDS2+ is a new and improved version of SMDS2. Samsung also offers support software that includes debugger, assembler, and a program for setting options.

SHINE

Samsung Host Interface for in-circuit Emulator, SHINE, is a multi-window based debugger for SMDS2+. SHINE provides pull-down and pop-up menus, mouse support, function/hot keys, and context-sensitive hyper-linked help. It has an advanced, multiple-windowed user interface that emphasizes ease of use. Each window can be sized, moved, scrolled, highlighted, added, or removed completely.

SAMA ASSEMBLER

The Samsung Arrangeable Microcontroller (SAM) Assembler, SAMA, is a universal assembler, and generates object code in standard hexadecimal format. Assembled program code includes the object code that is used for ROM data and required SMDS program control data. To assemble programs, SAMA requires a source file and an auxiliary definition (DEF) file with device specific information.

SASM88

The SASM88 is an relocatable assembler for Samsung's KS86-series microcontrollers. The SASM86 takes a source file containing assembly language statements and translates into a corresponding source code, object code and comments. The SASM86 supports macros and conditional assembly. It runs on the MS-DOS operating system. It produces the relocatable object code only, so the user should link object file. Object files can be linked with other object files and loaded into memory.

HEX2ROM

HEX2ROM file generates ROM code from HEX file which has been produced by assembler. ROM code must be needed to fabricate a microcontroller which has a mask ROM. When generating the ROM code (.OBJ file) by HEX2ROM, the value "FF" is filled into the unused ROM area up to the maximum ROM size of the target device automatically.

TARGET BOARDS

Target boards are available for all KS86-series microcontrollers. All required target system cables and adapters are included with the device-specific target board.

OTPs

One times programmable microcontrollers (OTPs) are under development for KS86C6308 microcontroller.

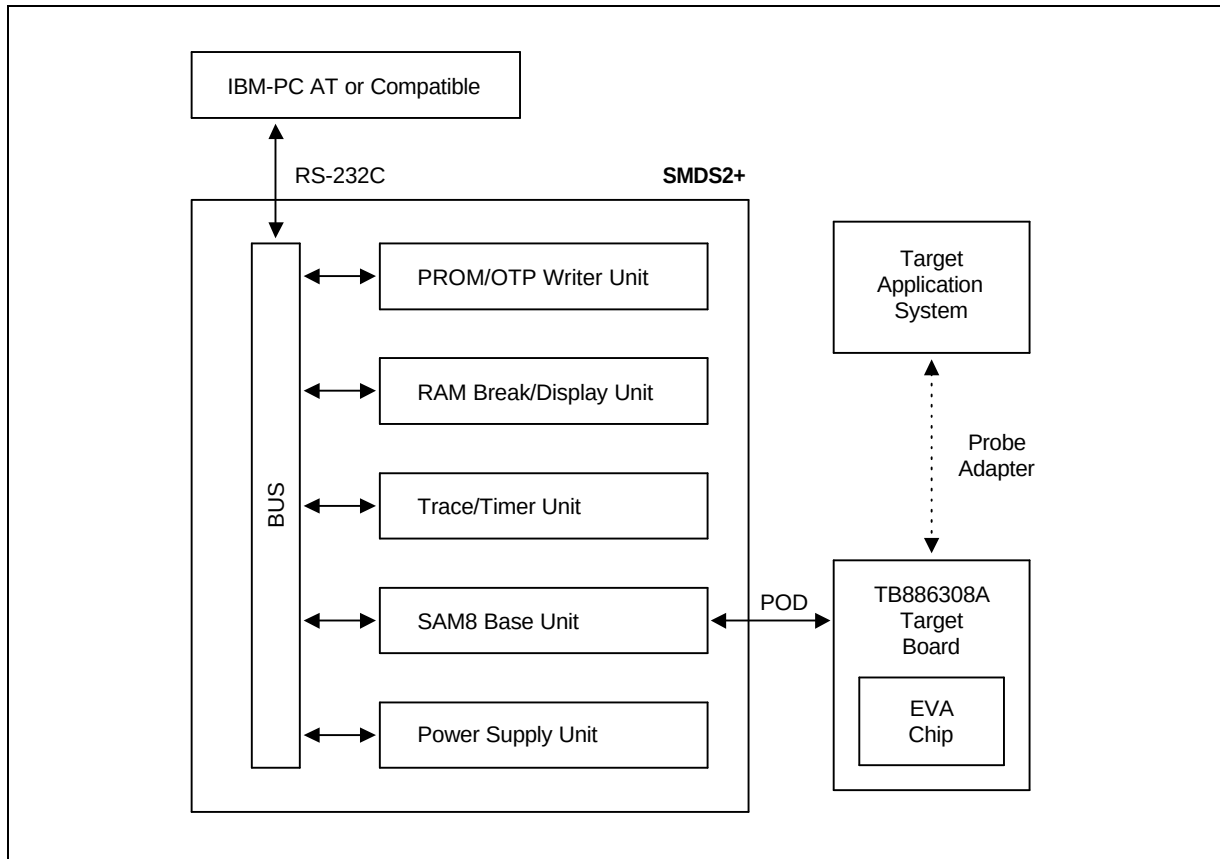


Figure 15-1. SMDS Product Configuration (SMDS2+)

TB866308A TARGET BOARD

The TB866308A target board is used for the KS86C6308 microcontrollers. It is supported by the SMDS2+ development systems. The TB866308A target board can also be used for KS86C6308.

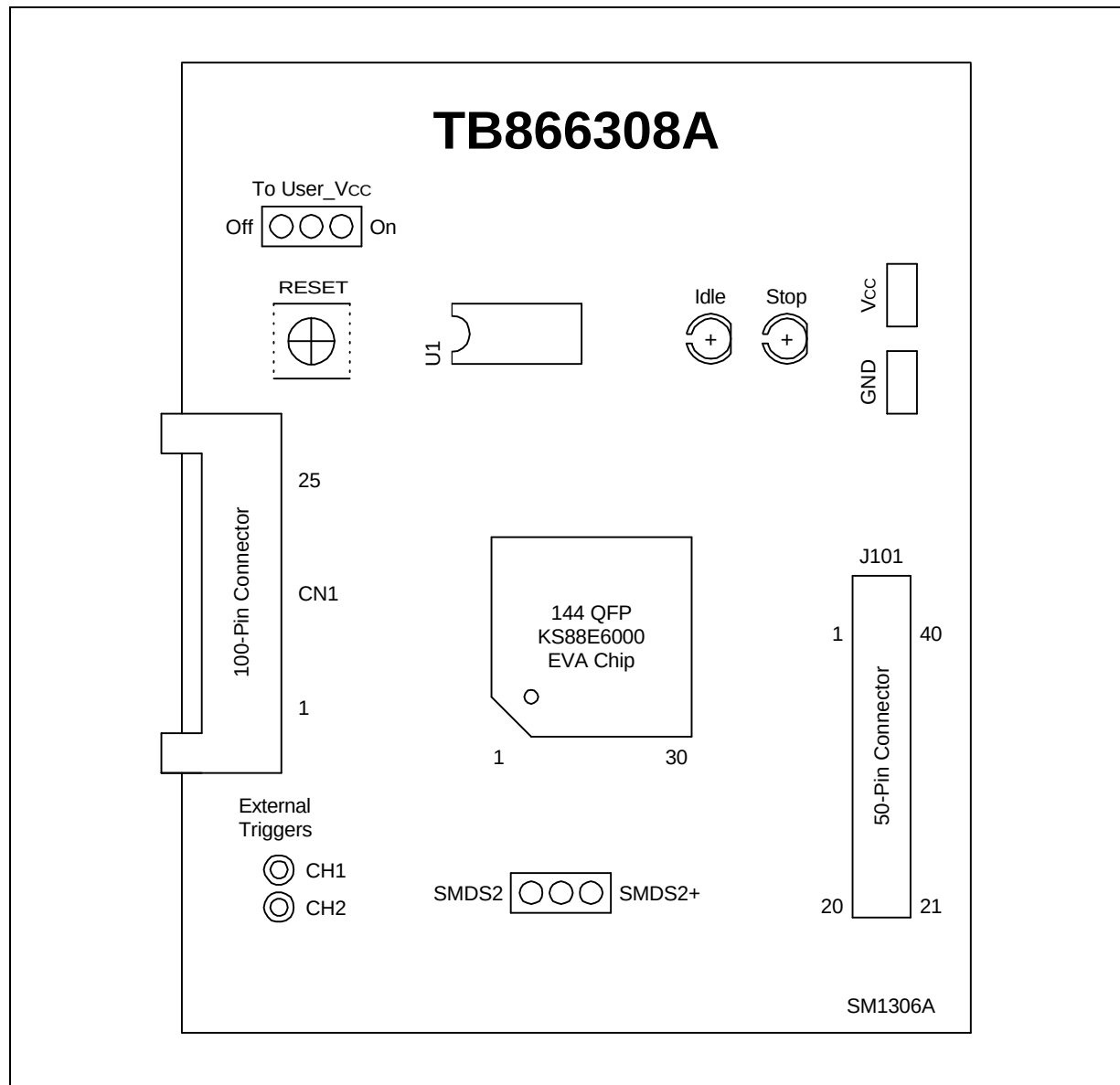
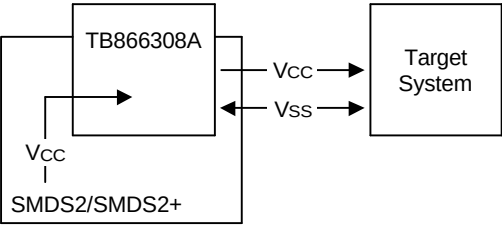
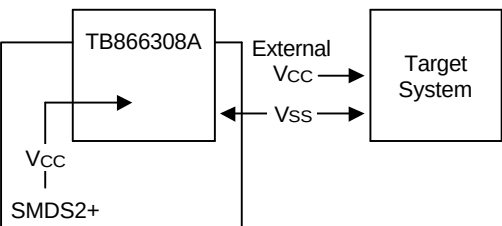


Figure 15-2. TB866308A Target Board Configuration

Table 15-1. Power Selection Settings for TB866308A

'To User_Vcc' Settings	Operating Mode	Comments
To User_Vcc Off  On		The SMDS2/SMDS2+ supplies V _{CC} to the target board (evaluation chip) and the target system.
To User_Vcc Off  On		The SMDS2/SMDS2+ supplies V _{CC} only to the target board (evaluation chip). The target system must have its own power supply.

NOTE: The following symbol in the "To User_V_{CC}" Setting column indicates the electrical short (off) configuration:



SMDS2+ Selection (SAM8)

In order to write data into program memory that is available in SMDS2+, the target board should be selected to be for SMDS2+ through a switch as follows. Otherwise, the program memory writing function is not available.

Table 15-2. The SMDS2+ Tool Selection Setting

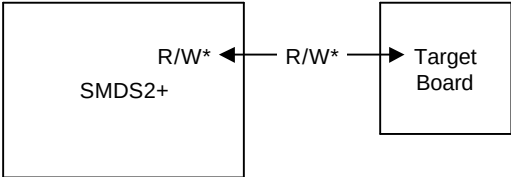
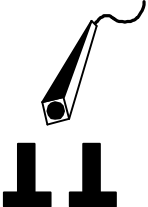
"SW1" Setting	Operating Mode
SMDS2  SMDS2+	

Table 15-3. Using Single Header Pins as the Input Path for External Trigger Sources

Target Board Part	Comments
External Triggers Ch1 Ch2	 Connector from external Trigger sources of the application System You can connect an external trigger source to one of the two external trigger channels (CH1 or CH2) for the SMDS2+ breakpoint and trace functions.

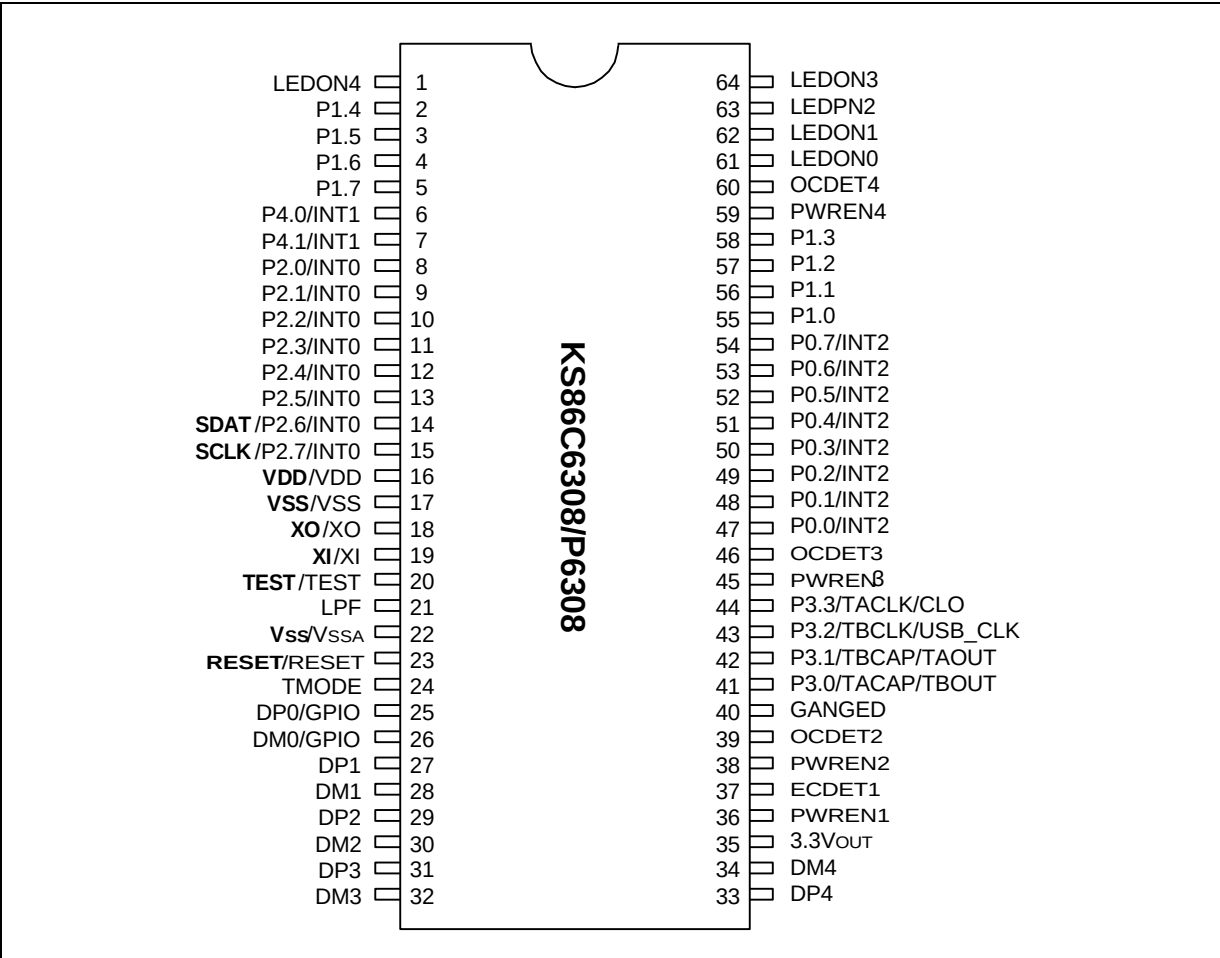


Figure 15-3. 64-Pin Connector for TB866308A

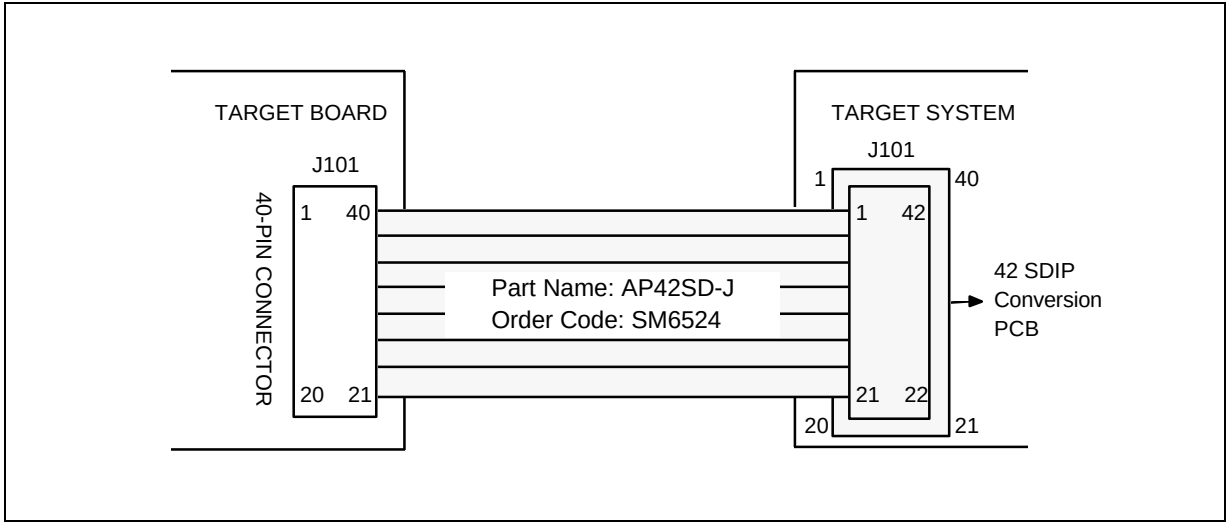


Figure 15-4. KS86C6308 Probe Adapter Cable for 64-SDIP Package